Méthodes d'optimisation pour l'espace et l'environnement

Thierry Touya

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *l'Université Toulouse III - Paul Sabatier*
Discipline ou spécialité : *Mathématiques apppliquées*

Présentée et soutenue par *Thierry TOUYA*

Le *19/09/2008*

Titre :

METHODES d'OPTIMISATION pour l'ESPACE et l'ENVIRONNEMENT

JURY

Didier AUROUX	*Maître de conférences*	*Université de Toulouse III*	*Co-directeur*
Yann CAILLOCE	*Ingénieur*	*Thalès Alenia Space*	*Examinateur*
Christian LE GALLIC	*Ingénieur*	*Centre d'Etudes de Gramat*	*Président*
Mohamed MASMOUDI	*Professeur*	*Université de Toulouse III*	*Directeur*
Bijan MOHAMMADI	*Professeur*	*Université de Montpellier*	*Rapporteur*
Bernard ROUSSELET	*Professeur*	*Université de Nice*	*Rapporteur*

Ecole doctorale : Mathématiques Informatique Télécommunications
Unité de recherche : Institut de Mathématiques de Toulouse - Equipe MIP
Directeur(s) de Thèse : Mohamed MASMOUDI - Didier AUROUX

À Alexandra, Mateo-Luis et Felix,
à mes parents,
à mes proches.

À ceux qui m'ont vu grandir
et à ceux que je vois grandir...

Avant-propos

Cette thèse a été réalisée à l'Institut de Mathématiques de Toulouse (IMT), au laboratoire MIP (Mathématiques pour l'Industrie et la Physique), au sein de l'équipe Optimisation de formes.
La première partie a été effectuée dans le cadre du projet OTOP (Optimisation Topologique pour des équipements-clé des télécommunications du futur) sélectionné par l'ANR (Agence Nationale pour la recherche), en collaboration avec l'entreprise Thalès Alenia Space.
La deuxième partie résulte d'un contrat avec la DGA (Délégation Générale pour l'Armement), en collaboration avec le Centre d'Etudes de Gramat (CEG).

Remerciements

J'ai ressenti un mélange de fierté et de soulagement lorsque j'ai compris que la thèse était vraiment finie : le jour où j'ai amené, seul, mon manuscrit au fond d'un sous-sol de reprographie. Dans cette précipitation euphorique, j'ai failli faire imprimer ce travail sans les remerciements. Cela aurait ôté la saveur de la thèse, étant donné que c'est la seule partie qui sera lue par tous mes proches et par la majorité des chercheurs qui par hasard tomberont dessus...

À tout seigneur tout honneur, je remercie chaleureusement en premier lieu mon directeur de thèse Mohamed Masmoudi : grâce à son sens du risque, j'ai pu m'initier malgré un profil atypique au monde mystérieux de la recherche.
Je te suis reconnaissant pour la passion et l'intelligence avec laquelle tu transmets ta vision profonde des objets mathématiques que nous manipulons parfois avec peu de précautions, pour ta connaissance de la science et au-delà, partagée autour de cafés animés qui réchauffent les neurones et pour tes conseils avisés.
Merci de m'avoir fait confiance et de m'avoir permis de mener à terme ce projet.

Je suis honoré d'avoir été également encadré par Didier Auroux : je suis son premier thésard, sur lequel il a pu se faire les dents (de lait). Chez lui, la gentillesse, une bonne humeur inoxydable, sans oublier une bienveillance décontractée, contrastent avec une rigueur, une intelligence et une efficacité hors normes. J'ai une théorie : en fait il y a quatre Didier et il s'arrange avec des ruses pour n'en présenter toujours qu'un seul à la fois en public... Je te sais gré d'avoir eu le tact de ne pas profiter de notre différence d'âge pour en faire une source de plaisanteries sans fin. Je m'incline devant ton sens du débat contradictoire, ta connaissance pointue des catastrophes aériennes et du championnat de foot. Resteront gravés tes facéties quotidiennes, la soirée de clôture du congrès SMAI 2007, les limonades sur le port de Baltimore ou au bord du lac d'Hossegor, le marathon de New-York que j'ai eu plaisir à courir avec toi.

J'adresse également toute mon estime et ma reconnaissance à Bijan Mohammadi et Bernard Rousselet pour avoir bien voulu accepter la charge de rapporteur, pour avoir eu le courage de lire ce manuscrit jusqu'au bout et avoir apporté critiques et commentaires sur ce travail.

Les autres membres du jury méritent tous mes égards : Christian Le Gallic, un président pour qui le rôle semblait taillé sur mesure et Yann Cailloce qui a participé à ma soutenance. En tant que collègues de travail, ils m'ont fait découvrir le métier d'ingénieur et donné du grain à moudre avec des problèmes passionnants.

Outre Christian, je témoigne ma gratitude à Maryse Vaullerin et Thierry Lacaze au Centre d'Etudes de Gramat (CEG) qui ont participé à ce travail. Au début nous étions un peu intimidés par les gendarmes et les posters aux murs décrivant l'activité du CEG. Mais ensuite, se perdre dans les villages aux alentours, prendre un café sur l'aire jardin des causses du Lot avec Mohamed furent parmi les meilleurs souvenirs de cette thèse. Un résultat important que nous avons réussi à établir est qu'il y a exactement quatre tunnels sur l'A20 entre Toulouse et le CEG (mais cette information classifiée ne peut malheureusement pas être rendue publique).

Outre Yann, je gratifie également de toute ma considération Gérard Caille à Thalès Alenia Space : ils ont initié cette thèse et ont toujours eu une antenne ou un satellite sous la main pour nous faire rêver.

Je souligne la contribution des autres chercheurs que j'ai eu l'occasion de rencontrer lors des réunions de projet, notamment les ingénieurs lyonnais d'Ansys. Je tire la révérence à Manu, avec qui j'ai commencé à apprendre les boucles "for" lors de séances de TP qui ont scellé notre destin.

Je rend hommage aux collègues enseignants qui m'ont appris le métier, m'ont souvent aidé avec sollicitude et qui exercent avec compétence : Sandrine Scott, Alain Huard, Olivier Mazet, Hélène Milhem à l'INSA, Maryline Marty-Guilhaumon et Agnès Boy-Dalverny à l'ENIT.

In memoriam : à mon ordinateur portable, qui pour cause d'utilisation intensive a vieilli prématurément et a succombé à la fin la thèse. Il eu une belle vie.
J'ai une dette aussi envers les logiciels de calcul, latex, Neil Young, Les Sopranos, Michael Connelly et tous les autres...

Aux collègues d'optimisation qui se sont intéressés à mon travail et à mon parcours : Sophie Jan et Frédéric Messine, votre soutien a compté.

Je dis toute mon amitié aux collègues d'équipe avec qui j'ai eu des affinités particulières et j'ai pu partager de très bons moments à la fois dans le labo et à l'extérieur : Jérôme et Mathieu F.

Je me sais redevable aux différents membres du labo qui animent le quotidien, Christine et Marie-Louise pour leur bonne humeur et énergie inébranlable face aux problèmes administratifs ingérables pour le commun des mortels, David pour son aide informatique, Monique pour sa gentillesse et pour savoir nous faire revenir sur Terre.

Je salue les autres thésards ou post-docs, soutiers de la recherche avec qui j'ai partagé des préoccupations communes et qui m'ont apporté une ouverture enrichissante : mes co-bureaux Liang, Mathieu F., Sébastien, puis Édith, Yogesh et Sheetal, Michaël, Yves, Jeremy à l'INSA et tous les autres.
Grâce à mon compagnon de travail Mathieu C. les journées sont plus agréables.

Je me félicite d'avoir été soutenu par mes parents, mes amis et ma famille lors de retrouvailles mémorables. J'ai bien senti au fil du temps, que seuls les plus courageux de mes proches s'aventuraient à me demander où j'allais enseigner cette année, quel était mon thème de recherche et dans quel but... D'ailleurs ça continue. Qu'ils m'aient considéré comme un Don Quichotte ou comme un matheux de plus qui décompense, ils ont continué à m'apporter confiance et amitié.

Chapeau bas à Jean-Jacques et Lionel pour le temps consacré à la tâche ingrate de relire mon manuscrit : vos remarques scientifiques et corrections m'ont été précieuses.

Mes pensées les plus fortes vont à Alexandra, Mateo-Luis qui m'ont accompagné au quotidien depuis le début de cette aventure et à Felix qui est arrivé au milieu. Merci de m'avoir entouré par monts et par vaux...

Table des matières

Introduction générale

Processus d'optimisation

Une multitude de problèmes (conception, planification, contrôle de processus, analyse de risque,...) dans de nombreux domaines de l'ingénierie demandent une **modélisation numérique** et un traitement sous forme d'optimisation.

L'**optimisation** est ici définie comme le moyen de **trouver la meilleure solution soumise aux conditions du problème** et on peut considérer que le processus s'effectue en trois étapes.

Première étape

Elle consiste à **analyser le problème** en opérant un certain nombre de choix :

- les paramètres intéressants à faire varier, ou **variables** du problème (comme la forme, les conditions, les matériaux,...),
- les limites pour leurs variations, ou **espace de recherche**, car la plupart du temps ces paramètres sont soumis à des contraintes de conception (physiques, économiques, techniques,...) ou de modélisation (implémentation),
- les **critères** numériques du problème, à maximiser (profit, qualité, rigidité,...) ou **minimiser** (coût, pertes, risque, poids, consommation,...) par rapport aux variables.

Deuxième étape

Il faut ensuite écrire une formulation mathématique, ou **modélisation** du problème.

On construit une **fonction coût** (ou fonction objectif) qui doit exprimer la **pertinence des solutions** par rapport aux désirs de l'utilisateur.

Sa définition peut être simplement analytique, mais souvent elle fait appel à un modèle numérique du dispositif étudié.

On obtient alors un **problème d'optimisation globale sous contraintes**, et il reste à choisir une bonne méthode mathématique pour le résoudre.

Souvent, l'objectif est de trouver une bonne solution admissible, et dans l'idéal, on cherche la meilleure solution admissible, c'est à dire la solution optimale globale.

La modélisation de systèmes complexes rend généralement le problème d'optimisation de la fonction coût très difficile :

- Ces modèles sont non linéaires, souvent bruités, et avec un grand nombre de variables.

  Ils engendrent de nombreux minima locaux : or les méthodes mathématiques d'optimisation les plus efficaces soit concernent des problèmes plus réguliers, soit convergent vers le premier minimum local.

- Les évaluations de la fonction sont coûteuses en ressources : cela écarte les méthodes d'ordre zéro (sans gradient) qui nécessitent un nombre élevé d'évaluations.

- La fonction est non différentiable (ou le calcul du gradient est difficile à mettre en oeuvre) : or les autres méthodes d'optimisation utilisent le gradient.

Troisième étape

Il reste à **synthétiser les solutions** proposées qui sont évaluées, puis éventuellement éliminées, jusqu'à obtention d'une solution ou d'une méthode acceptable.

Le coeur de notre travail concerne la deuxième étape, les autres demandant l'expertise et le jugement des ingénieurs qui posent le problème initial.

Organisation de la thèse

D'autre part, cette thèse se compose de **deux études indépendantes**, avec deux **approches différentes** des problèmes d'optimisation.

Première partie

Elle est issue d'une collaboration industrielle qui porte sur la **conception d'une antenne spatiale** réseau active.

Le **premier problème** est de **calculer les lois d'alimentation optimales** : **nous disposons** d'une fonction de rayonnement explicite, et donc **des gradients**.

La difficulté provient ici du **grand nombre de minima locaux** : nous **reformulons le problème** par convexification afin d'obtenir **un seul minimum global**, qui peut être trouvé avec une méthode efficace.

Le **deuxième problème** consiste à **réduire le nombre d'éléments rayonnants** tout en conservant les performances de l'antenne : c'est un problème d'**optimisation topologique**.

La difficulté réside ici dans la modélisation du problème : il faut trouver un critère à optimiser.

Nous construisons une fonction coût après avoir **extrait une information** de l'ensemble des **lois optimales** : un **algorithme de type gradient topologique décide les regroupements** entre éléments rayonnants élémentaires.

Deuxième partie

Elle porte sur une **simulation de dispersion** de produit dans l'environnement : nous effectuons une **étude de fiabilité et de sensibilité**.

Le **premier problème** est le calcul de **probabilités de défaillance**.

Le **deuxième problème** est la recherche du **point de conception**, qui est un problème d'**optimisation**.

Le **troisième problème** est la **détermination des paramètres influents**.

On recherche ici une méthode valable pour un grand nombre de paramètres incertains et pour une fonction définie par une suite de programmes fonctionnant en boîte noire.

La difficulté provient du **coût de calcul en grande dimension** et du fait que le **gradient n'est pas disponible**.

Nous utilisons l'approche par **approximation du modèle** (méthode de surface de réponse), en comparant les résultats numériques :

- pour une méthode d'apprentissage régularisé de **réseau de neurones**,
- et pour une méthode d'interpolation sur grille éparse (***sparse grid***) avec algorithme adaptatif.

  Nous améliorons les résultats en grande dimension grâce à une méthode d'**approximations successives** et une autre de **seuillage des données**.

Première partie

Contrôle et optimisation topologique d'antennes spatiales

Introduction

Le travail de l'ingénieur consiste souvent à trouver la meilleure performance de son produit avec le souci du coût minimum : c'est une illustration du concept d'optimisation, dans le sens de chercher la meilleure solution à un problème technique en tenant compte des contraintes industrielles.

Au laboratoire MIP (pour Mathématiques pour l'Industrie et la Physique) de l'IMT (pour Institut de Mathématiques de Toulouse), sous la direction de Mohamed Masmoudi et Didier Auroux, notre rôle a été d'apporter une expertise en résolution de problèmes d'optimisation mathématique au service d'un projet de recherche industriel en collaboration avec l'entreprise **Thalès Alenia Space** (ex Alcatel).

Cette étude a été retenue par l'**ANR** (pour Agence Nationale de la Recherche), section **RNRT** (pour Réseau National de la Recherche Telecom), dans le cadre d'un projet nommé **OTOP** (pour Optimisation Topologique pour les équipements clé des télécommunications du futur).

Nous avons participé à la partie "Antennes" du projet, les autres partenaires principaux pour la partie "Composants" sont la société France Telecom et le CNES (pour Centre National des Etudes Spatiales).

Le projet porte sur la **conception d'une antenne réseau spatiale active** : l'objectif est de trouver une méthode permettant de calculer une géométrie d'antenne qui **minimise le nombre d'éléments rayonnants** pour des raisons de coût, tout en satisfaisant des **performances de rayonnement**. Pour une antenne donnée, il faut aussi savoir calculer les lois d'alimentation des éléments rayonnants qui permettent d'obtenir ces performances.

Dans cette étude, nous avons mobilisé des connaissances dans le domaine de l'optimisation numérique et de l'optimisation topologique (dans les deux cas nous nous sommes ramenés à la minimisation d'une fonction coût).

La première partie de ce manuscrit est organisée de la manière suivante :

- **Introduction générale**

  Nous présentons d'abord les **concepts** liés aux antennes étudiées, qui sont utilisées pour des missions multimedia par satellite.

  Leur structure est un **réseau régulier d'éléments rayonnants contrôlés en amplitude et en phase** qui permet de **dépointer le faisceau successivement** sur une série de spots sur la Terre qui forment la couverture de l'antenne : l'alimentation de l'antenne peut être reconfigurée à chaque instant pour changer de zone.

Les contraintes de rayonnement exigées sont présentées : elles induisent un nombre excessif d'éléments qu'il faut réduire le plus possible tout en conservant les performances de l'antenne de départ.

La modélisation complète du problème est ensuite présentée.

Puis nous rappelons des méthodes générales de l'optimisation numérique afin d'éclairer le choix de l'algorithme de utilisé pour le calcul des lois d'alimentation.

Enfin nous exposons les méthodes de l'optimisation topologique qui ont guidé le choix de l'algorithme de calcul de la nouvelle géométrie de l'antenne.

- **Optimisation des lois d'alimentation**

  Pour une géométrie d'antenne et un spot donnés, nous détaillons la formulation mathématique du problème de calcul des lois optimales : il faut **vérifier** des **contraintes de rayonnement** et une **contrainte sur les modules**.

  C'est un problème d'optimisation difficile, connu pour avoir un **grand nombre de minima locaux**.

  Notre approche comporte deux **idées fondamentales** :

  - La première est de **réduire la non-linéarité** du problème en choisissant comme **variables d'optimisation la partie réelle et imaginaire** des composantes de l'alimentation complexe au lieu de prendre classiquement le **module et la phase**.

    Pour la même raison, nous ne convertissons pas en décibels les données du problème.
  - La deuxième consiste à formuler un **problème numériquement équivalent plus simple car convexe : sa solution unique est le minimum global du problème initial**.

  Nous exposons ensuite la construction de la fonction coût et l'algorithme de minimisation de type Levenberg-Marquardt utilisé : nous disposons ici d'une fonction explicite et de son gradient.

  Cette **méthode de calcul des lois d'alimentation, inédite, constitue un algorithme rapide et robuste**.

  Les résultats de tests numériques sont présentés pour deux études différentes.

- **Optimisation topologique de l'antenne** :

  Il n'existe pas de méthode générale pour **réduire le nombre de sources** d'une antenne réseau active.

  Pour construire la nouvelle géométrie, l'idée de départ est de **regrouper les éléments rayonnants identiques** du réseau initial et former un réseau irrégulier avec moins d'éléments.

  Chaque regroupement est relié à un seul dispositif électronique de contrôle : la diminution du nombre de sources constitue ainsi une amélioration de conception cruciale.

  Les difficultés de ce problème de *design* sont :

- gérer l'agencement d'un nombre limité de formes de regroupements pour des raisons industrielles,
- trouver une géométrie unique qui conserve des performances similaires à celles de l'antenne initiale, et valables pour l'ensemble des spots (à chaque spot correspond une loi d'alimentation différente), avec moins de degrés de liberté.

Nous avons appliqué deux **idées majeures** :

- La première est de calculer les lois d'alimentations optimales pour l'antenne initiale et pour l'ensemble des spots en **supprimant la contrainte sur les modules** : cela permet d'avoir un **degré de liberté supplémentaire** pour respecter les contraintes de rayonnement avec de la marge.

  Une **technique de réduction de données**, la décomposition en valeurs singulières, nous permet ensuite d'extraire un **résumé des modules optimaux** précédents.
- La deuxième est d'obtenir la **nouvelle géométrie** lors d'un processus itératif, où les **regroupements successifs** font **diminuer une fonction coût** construite à partir de l'information précédente, avec un algorithme de type **gradient topologique**.

Nous appliquons ensuite la méthode à une étude correspondant à une mission satellite classique : le **nombre d'éléments** de l'antenne initiale a été **divisé par deux pour des performances équivalentes**, ce qui constitue un résultat nouveau et satisfaisant.

Dans le cadre du projet ANR, ces algorithmes ont donné lieu à la **création** du **logiciel opérationnel OTOP** (coopération avec la société Ansys).

Chapitre I

Introduction au problème

Sommaire

1 Motivation

Télécommunications Une antenne est un dispositif capable de transmettre ou de recevoir des signaux par des ondes électromagnétiques. Les premières études datent du XIXe siècle, avec Hertz et Marconi qui ont réalisé des transmissions radio en application des équations de Maxwell.

Un avancement significatif dans le développement des antennes s'est produit durant la seconde guerre mondiale à cause de leur importance militaire stratégique dans le domaine des radars et des télécommunications.

Le secteur des télécommunications connaît à partir de ce moment là une expansion continue :

- Dans les années soixante commence l'ère spatiale avec les communications téléphoniques et la diffusion de la télévision intercontinentale par satellite.

- Dans les années soixante-dix, la numérisation permet de véhiculer simultanément plusieurs communications sur une même ligne, et assure également l'intégration des services, en transmettant des informations de nature différente : voix, images, écrits, données.
- Les années quatre-vingt voient la naissance de la téléphonie mobile et de l'internet.

Le rythme de l'innovation s'est alors accéléré : il a provoqué la convergence des télécommunications, de l'informatique et de l'audiovisuel, pour parler aujourd'hui des NTC (pour Nouvelles Technologies de la Communication) et de l'avènement de la société de l'information.

Ce secteur concerne plusieurs domaines scientifiques complémentaires :

- les mathématiques avec le traitement du signal, la cryptographie, la théorie de l'information et la modélisation numérique,
- la physique avec l'électromagnétisme, les semi-conducteurs, l'électronique et l'opto-électronique,
- l'informatique avec le génie logiciel et la diffusion de la micro-informatique.

Les progrès technologiques ont fait baisser les coûts et entraîné la démocratisation de leur usage : aujourd'hui, ces techniques ont pénétré notre espace professionnel et notre espace privé, et on cherche à communiquer toutes sorte de données depuis n'importe quel endroit avec un débit de transmission toujours plus élevé pour garantir l'interactivité des applications.

Pour faire face à l'augmentation du nombre des utilisateurs et des débits, les futurs réseaux de communications devront mettre en oeuvre des techniques de plus en plus évoluées.

Antennes satellitaires Parmi les approches possibles, les communications par satellite sont actuellement essentielles pour permettre cette grande circulation d'informations à l'échelle planétaire.

Les principales qualités pour un système efficace, du point de vue de l'antenne satellitaire, sont :

- Une haute performance : l'objectif est de concentrer l'énergie rayonnée dans une direction précise de l'espace, ce qui est fondamental pour la communication à longue distance.
- Une aire de couverture bien définie : pour éviter des interférences avec les aires voisines.
- La robustesse : le système doit fonctionner pendant une quinzaine d'années (durée de vie moyenne d'un satellite commercial) sans possibilité d'intervenir en cas de panne.
- La reconfigurabilité : c'est-à-dire la capacité de modifier le diagramme de rayonnement en intervenant sur l'alimentation.

Les **antennes réseaux actives** (ou antennes intelligentes) sont une solution : elles allient performance, avec une amélioration de la capacité et de la qualité de transmission, et souplesse. Leur principe est de combiner un **réseau** (linéaire, circulaire, plan,...) d'**éléments rayonnants** et un **dispositif électronique de contrôle**.

Un logiciel, qui peut être apte à répondre de manière dynamique aux spécifications désirées, modifie l'alimentation et donc le rayonnement. Pour cela, on utilise un processeur numérique de traitement du signal, qui en combinant les signaux de l'ensemble des sources élémentaires, permet de former un ou plusieurs diagrammes de rayonnement. Pour le contrôle et la formation de ces diagrammes, il faut appliquer une loi d'alimentation électrique sur les éléments rayonnants correspondant aux critères fixés (maximisation du gain dans une direction donnée, maîtrise du niveau des lobes secondaires,...).

Les algorithmes de formation de faisceaux vont permettre d'émettre dans des directions particulières avec un gain maximum (voir par exemple [HT02]), ce qui est l'objectif.

Cette reconfiguration facile du diagramme de rayonnement conduit en contrepartie à des coûts plus élevés de montage, à un contrôle plus difficile et à un poids plus élevé du système (voir [Bal97]).

Or dans la conception du produit, le coût de fabrication est un critère fondamental à optimiser pour l'industriel (par exemple le poids de l'antenne satellitaire doit être le plus faible possible en raison du coût élévé du lancement).

Collaboration industrielle Notre collaboration avec Thalès porte sur une antenne réseau active pour des applications multimedia par satellite : les performances exigées impliquent un très grand nombre de sources (plusieurs centaines). La couverture est multispot et demande de nombreux dépointages du faisceau (plusieurs dizaines) : il faut d'abord trouver un algorithme de recherche des lois d'alimentations efficace en grande dimension.

Pour concevoir l'antenne optimale, il faut ensuite pour des raisons de coût cruciales réduire le plus possible le nombre de sources tout en conservant les performances : cette réduction s'accompagnera du passage d'un réseau régulier à un réseau irrégulier, ce qui permettra d'abaisser les lobes de réseau.

L'objectif est de disposer d'un algorithme permettant la création d'un logiciel de conception.

2 Antennes

2.1 Antennes spatiales

2.1.1 Satellites de communication

Présentation Les satellites de communication permettent depuis les années soixante de développer les formes de communication modernes (téléphonie, radio, télévision) en relayant les signaux à l'échelle planétaire.
Ils ont connu une expansion très importante (nombre estimé à un millier) à cause

de l'importance du développement des communications et applications multimedia dans notre société de l'information.

Le satellite est équipé d'une antenne qui reçoit le signal d'une station terrestre : elle l'amplifie et le transmet à une station réceptrice en utilisant une autre longueur d'onde.

Le satellite peut être placé dans l'espace dans les situations suivantes :

- Il peut évoluer sur une orbite géostationnaire : il fait le tour de la Terre en 24 heures, à vitesse constante, à 36 000 km à la verticale de l'équateur et paraît immobile pour un observateur au sol.
  Les antennes au sol, qui doivent être pointées vers le satellite, peuvent ainsi fonctionner sans être équipées d'un système coûteux de poursuite des mouvements du satellite, et un seul satellite permet de couvrir une zone importante.
- Il peut évoluer sur une orbite basse : la mise en orbite et le satellite (qui demande une puissance de signal moins importante) sont moins coûteux, mais il bouge rapidement pour un observateur au sol.
  Un grand nombre de satellites est alors nécessaire (constellation satellitaire) pour assurer une connectivité permanente. On les utilise pour les systèmes de télédection, positionnement (systèmes GPS et Galileo) et téléphone satellitaire.

De plus, les satellites de communication améliorent les possibilités des moyens traditionnels (fibre optique, couverture radio) grâce aux avantages suivants :

- possibilité de relier facilement les continents entre eux,
- diminution du coût d'installation et de maintenance d'un réseau terrestre, qui augmente avec sa longueur.

Ils présentent ainsi une vocation d'utilisation à l'échelle mondiale, particulièrement adaptée aux zones géographiques isolées ou très vastes, pour lesquelles l'installation d'un réseau terrestre serait trop coûteuse : ils permettent de **réduire la fracture numérique** en permettant l'accès du plus grand nombre aux services multimédia.

La figure I.1 présente l'exemple d'une constellation de satellites pour une couverture multimedia mondiale (société Intelsat).

Antennes satellitaires De nos jours, le développement des moyens de communications interactifs (internet à réponse immédiate, vidéoconférence, télémédecine, téléenseignement,...) passe en priorité par l'utilisation d'une antenne satellitaire émettant dans la bande dite Ka (20-30 GHz).

Pour couvrir la zone terrestre souhaitée, l'antenne doit constamment être repositionnée : la méthode traditionnelle consiste à utiliser un mécanisme commandé par des moteurs électriques, mais dans le domaine spatial, on préfère éviter les solutions mécaniques, pour des raisons de fiabilité.

Les systèmes hyperfréquence du futur utiliseront donc, de manière générale, des antennes à balayage électronique, dont le fonctionnement est expliqué dans la section suivante.

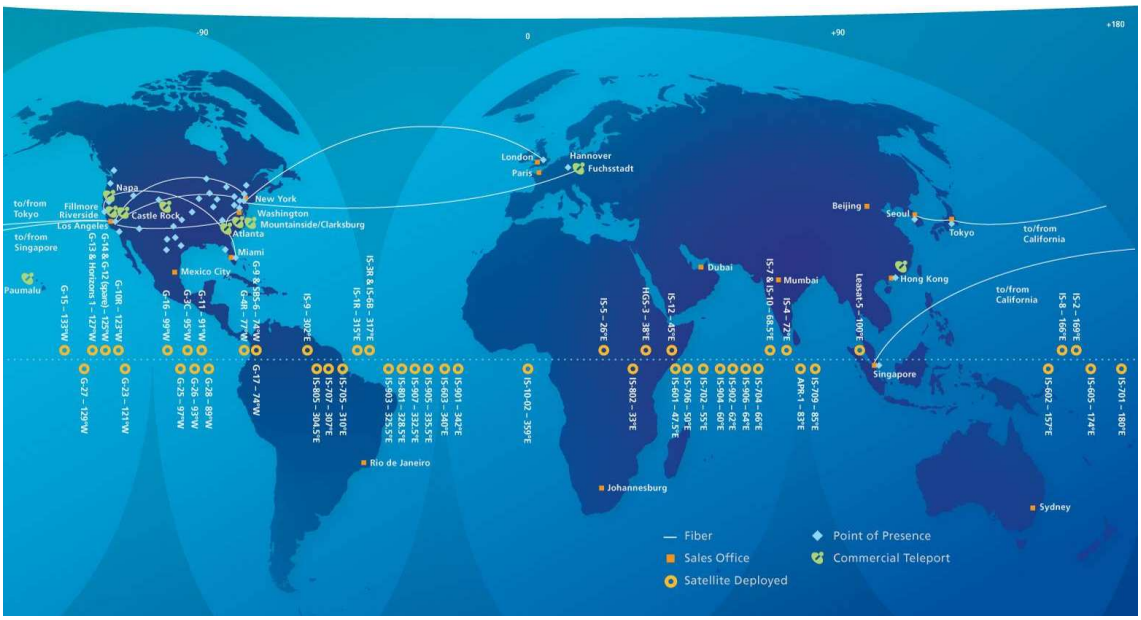


FIG. I.1 – Constellation de satellites Intelsat

Un besoin grandissant de ce type d'antennes apparaît, de plus elles présentent de nombreuses qualités :

- baisse des coûts des satellites (contrainte primordiale),
- performances accrues de ces antennes en terme de fiabilité, grâce à la possibilité de recalculer l'alimentation de l'antenne en cas de panne de certaines sources,
- totale flexibilité pour la formation des faisceaux (tant à l'émission qu'à la réception) et leur reconfiguration : le changement de direction est instantané,
- fonctionnement en mode multi-faisceaux,
- intégration des antennes dans les structures (antennes conformes).

2.1.2 Antennes réseaux actives

Eléments rayonnants (ER) Une source de rayonnement (un cornet très compact ou un ensemble de patches) est un dispositif électronique alimenté, ou contrôlé, uniformément en amplitude et phase et qui émet un champ électromagnétique.

Ces éléments rayonnants seront appelés dans la suite **ER**.

Réseau Nous nous intéressons aux antennes réseaux à rayonnement direct (DRA, pour *Direct Radiating Array*), ou antennes réseaux actives, constituées d'un ensemble d'ER disposés en réseau (voir figure I.2).

Les géométries les plus utilisées sont les réseaux linéaires et plans pour pouvoir concentrer la puissance, et les réseaux circulaires pour l'indifférence de leur orientation.

Le nombre d'ER varie entre deux et plusieurs milliers, leurs dimensions et espacements (typiquement $\frac{\lambda}{2}$ pour un réseau régulier, avec λ la longueur d'onde) varient avec la fréquence et l'application concernée (radio, radar, sonar, barrettes d'échographie,...).

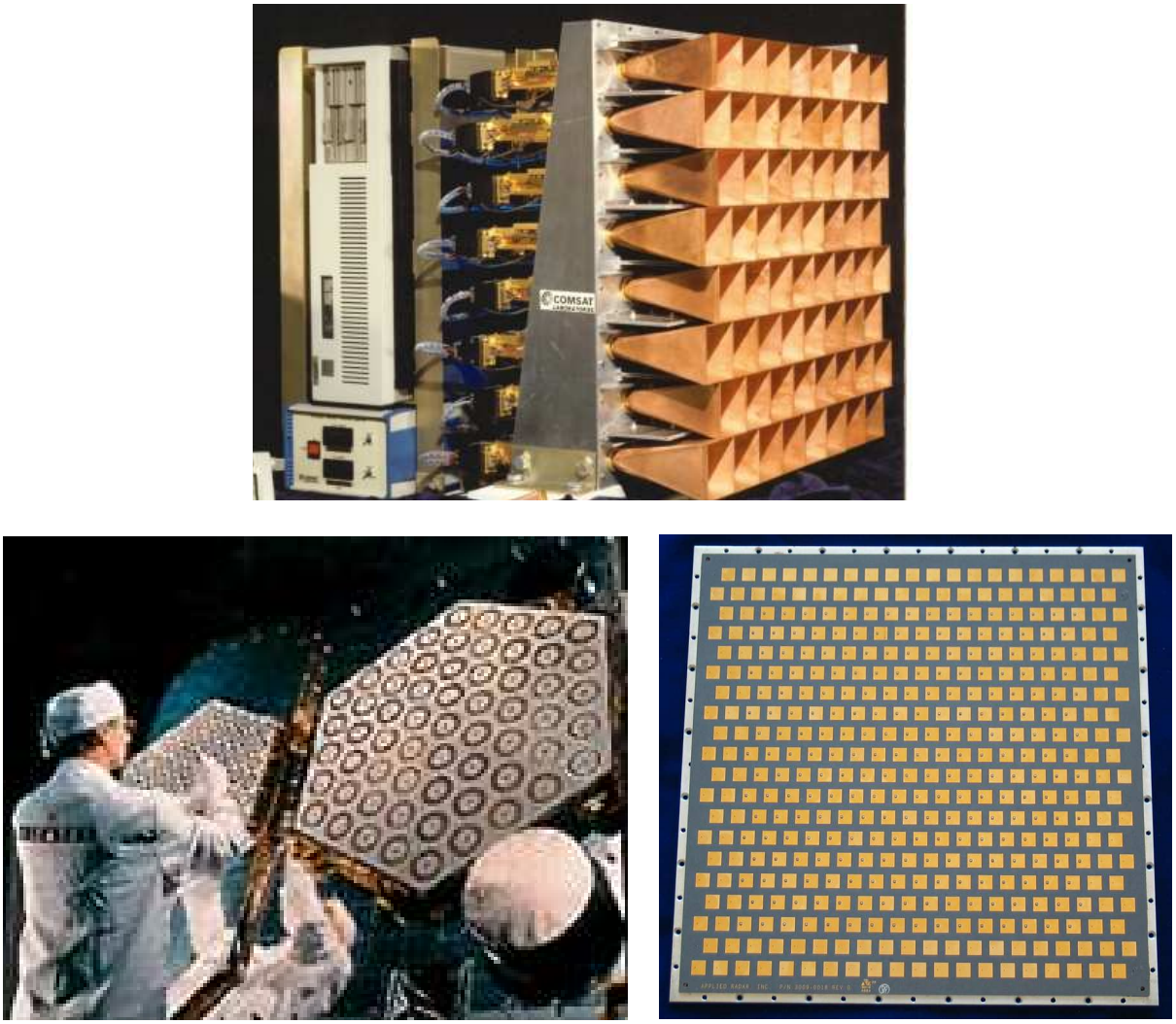


FIG. I.2 – Exemples d'antennes réseaux planes

Les contraintes technologiques actuelles imposent les antennes de type émission à réseau plan pour les applications de télécommunications multimedia par satellite, nous considérons uniquement ces antennes dans la suite.

Une solution classique pour les antennes multimedia satellitaires consiste à utiliser un grand réseau plan d'ER à maille régulière comme celui de la figure I.3. Nous considérons ce cas dans la suite.

2.1.3 Caractéristiques

Nous renvoyons le lecteur à [Tho90] ou [Dra86] pour des explications plus détaillées sur les caractéristiques des antennes.

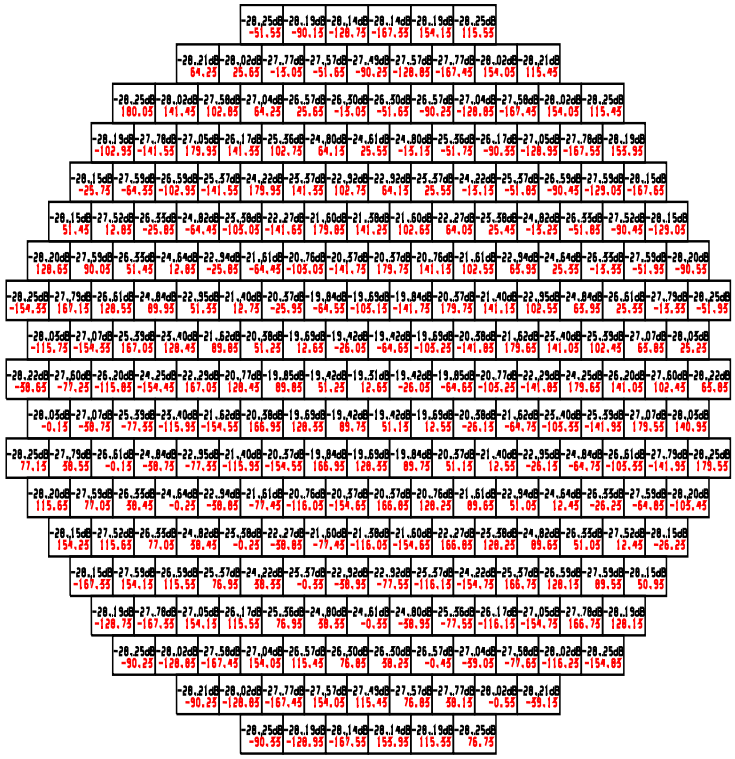


FIG. I.3 – Antenne réseau d'émission

Loi d'alimentation a La loi d'alimentation de l'antenne notée $\boldsymbol{a}$ est le vecteur complexe composé des alimentations de tous les ER du réseau.

Principe de Huygens Aux fréquences très élevées, il devient possible de grouper les ER afin que les interférences entre les champs rayonnés concentrent l'énergie dans une direction choisie.

En effet, d'après le principe de Huygens

- les rayonnements des ER se combinent et forment un faisceau,
- si l'alimentation est uniforme sur l'ensemble des ER, la direction du faisceau est perpendiculaire à la surface du support,
- pour dépointer le faisceau, il suffit de jouer sur les phases : on contrôle ainsi le rayonnement émis.

Champ E Le champ électromagnétique noté $\boldsymbol{E}$ émis par le réseau s'obtient comme combinaison linéaire des champs des ER et se calcule en une direction de l'espace.

Energie $|E|^2$ L'énergie, ou puissance, rayonnée par le réseau est $|E|^2$ le module du champ au carré.

Directivité $\mathcal{D}$ La directivité notée $\boldsymbol{\mathcal{D}}$, ou gain de l'antenne, est le quotient de la puissance rayonnée dans une direction par la puissance totale :
elle **mesure la capacité à concentrer l'énergie rayonnée dans une certaine direction**.

La directivité est égale à l'énergie à un facteur près. Elle peut s'exprimer en décibels (dB) : c'est une échelle logarithmique qui permet de comparer des valeurs d'ordres de grandeur très différents

En dépit de propriétés spécifiques bien particulières, une antenne réseau suit la règle simple selon laquelle la directivité est d'autant plus élevée que sa dimension est grande par rapport à la longueur d'onde.
Ainsi, on affecte des ondes longues aux applications pour lesquelles la directivité n'est pas essentielle et, inversement, des ondes courtes lorsque, par souci d'économie ou de discrétion, une certaine directivité est souhaitable.

Diagramme de rayonnement Le diagramme de rayonnement est la fonction qui caractérise la répartition du champ électrique. Dans la suite, nous nous plaçons dans le cas où le diagramme de rayonnement désigne la directivité.

Représentations graphiques Pour visualiser le diagramme de rayonnement en dB, on peut utiliser les représentations de la figure I.4 :

- vue en 3 dimensions,
- vue en deux dimensions avec courbes iso-niveaux,
- coupe suivant une direction.

Dans cette étude, nous utiliserons la deuxième représentation.

Lobes de rayonnement La direction du maximum de rayonnement fait apparaître un lobe principal qui contient la quasi-totalité de l'énergie rayonnée.
De part et d'autre de celui-ci apparaissent des lobes secondaires qu'on s'efforce souvent de réduire.

2.1.4 Conservation de l'énergie

Une relation fondamentale entre la loi d'alimentation a et le diagramme de rayonnement (dans un milieu supposé non absorbant) est la conservation de l'énergie.

Pour une puissance d'alimentation des ER constante, quand on modifie la loi d'alimentation, la puissance globale rayonnée est conservée. La réduction de la puissance dans certaines directions entraîne son augmentation dans d'autres et inversement.

2.2 Conception

2.2.1 Contraintes de rayonnement

Zones de gain, d'isolation, de couverture On définit sur la Terre deux types de zones pour les contraintes de rayonnement :

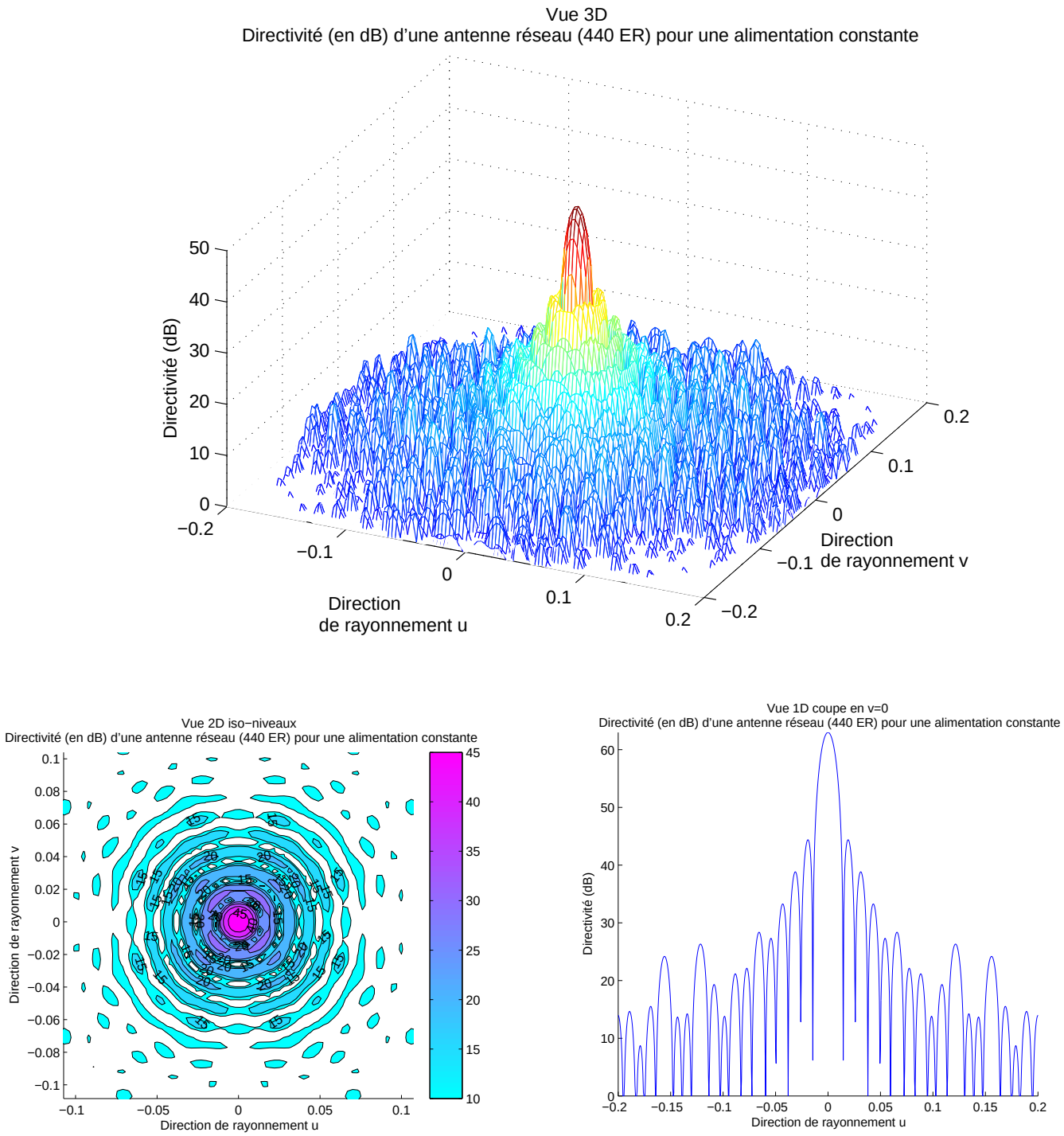


FIG. I.4 – Vues 3D, 2D, 1D du diagramme de rayonnement d'une antenne réseau (440 ER) pour une alimentation constante

- La **zone de gain** : dans cette zone, le **niveau de puissance rayonnée**, ou de directivité, doit être **le plus fort**.
  Elle correspond à la partie utile du rayonnement, le lobe principal.

- La **zone d'isolation** : dans cette zone, le **niveau de puissance rayonnée** doit être **faible**.

Elle correspond à la région où on ne veut pas émettre (par souci de confidentialité, pour une économie d'énergie, pour éviter les interférences...) et où apparaissent les lobes secondaires qu'on veut abaisser.

La **zone de couverture** est la région où on veut dépointer le faisceau de l'antenne sur plusieurs **zones de gain successives**, c'est la zone destinée au rayonnement principal de l'antenne, et elle peut être :

- simple : elle est constituée d'un ensemble de spots circulaires ou elliptiques adjacents,
- formée : les zones de gain épousent alors une zone géographique dont le contour est complexe, comme un pays par exemple.

Gabarit Le gabarit décrit la forme souhaitée du diagramme de rayonnement dans les diverses directions de l'espace, ou points sur la Terre.

On le donne sous forme de **valeurs de directivité à respecter sur les zones de gain et d'isolation**.

Problème de synthèse de réseau La synthèse de réseau consiste à **calculer la loi d'alimentation** d'une antenne réseau de façon à **respecter un gabarit** donné. On trouve les formes suivantes :

- Synthèse fixe : on veut déterminer un certain nombre de diagrammes de rayonnement, le temps de calcul n'est pas essentiel et on peut utiliser des méthodes complexes.
- Synthèse adaptative : la loi d'alimentation d'une antenne adaptative est mise à jour de façon permanente pour s'adapter au milieu, et les algorithmes doivent être très rapides.

Dans la suite nous nous plaçons dans le premier cas.

On trouve ensuite trois types de synthèses :

- Synthèse en amplitude seulement : elle permet de réaliser des lobes directifs symétriques mais les applications sont limitées.
- Synthèse en amplitude et en phase : elle permet de réaliser des lobes directifs avec des lobes secondaires fortement contrôlables. L'inconvénient est que sa mise en oeuvre s'effectue avec des composants coûteux.
- Synthèse en phase seulement : elle permet de réaliser des lobes directifs avec des lobes secondaires moins contrôlables que la précédente. Mais sa mise en oeuvre nécessite des déphaseurs d'un coût plus faible, et cette méthode semble un bon compromis.

2.2.2 Contraintes techniques

Contraintes de fabrication Selon le type d'antenne, on peut avoir des contraintes physiques sur ses éléments, par exemple la taille et la forme des ER.

Contraintes d'alimentation En ce qui concerne l'alimentation complexe de l'antenne :

- Les phases sont libres : elles servent à dépointer le signal.
- Les modules sont généralement soumis à des contraintes techniques.

Par exemple on doit respecter pour l'amplitude des modules :

- soit une valeur maximum (fonctionnement des amplificateurs),
- ou un écart maximum (pour minimiser les effets de couplage, c'est-à-dire les interférences de rayonnement entre les ER)
- ou une valeur fixée.

2.2.3 Programme de conception

Définition Le problème de conception d'une antenne réseau active se pose sous la forme suivante :

"Soit un gabarit, c'est-à-dire une fonction diagramme de rayonnement désiré D, quelles sont la géométrie du réseau et la loi d'alimentation optimales qui permettent d'approcher D ?".

Le cycle général de conception est alors :

1. choix des caractéristiques physiques des ER : taille, forme, (et par exemple nature et épaisseur du substrat diélectrique,...),
2. choix de la géométrie du réseau : c'est-à-dire nombre et position des ER,
3. calcul des lois d'alimentation optimales,
4. validation expérimentale.

Détaillons les étapes 2 et 3 du cycle de conception précédent.

Géométrie de l'antenne : optimisation topologique On veut donc trouver la meilleure géométrie du réseau qui réalise les performances attendues.

Synthèse de réseau : optimisation de la loi d'alimentation Etant donné une antenne à géométrie fixée et un gabarit, effectuer la synthèse de réseau consiste à calculer la loi d'alimentation des ER, dite loi d'alimentation optimale, respectant les contraintes de gabarit et d'alimentation.

Découplage des problèmes Quelques méthodes de synthèse effectuent les étapes 2 et 3 du cycle de conception à la fois : la synthèse du réseau en jouant non seulement sur la loi d'alimentation mais aussi sur la géométrie du réseau (disposition des ER dans le cas de la méthode des perturbations dans [Che71]). Mais la plupart des méthodes se déroulent à géométrie du réseau fixée.

Dans la suite, nous découplons les problèmes et nous traitons les deux problèmes d'optimisation (des lois et de la géométrie) séparément.

2.3 Collaboration industrielle

Introduction Cette étude est basée sur une collaboration entre l'entreprise Thalès Alenia Space et le laboratoire MIP de l'IMT.

Les ingénieurs du secteur recherche amont de Thalès pour les antennes satellitaires ont soumis au MIP deux études d'optimisation d'antenne :

- une étude de faisabilité sur cas test pour dégager une méthode originale,
- une étude industrielle pour mettre la méthode en pratique sur un cas de mission réaliste et concevoir un logiciel.

Nous détaillons dans la suite plus précisément le contexte de cette collaboration.

Thalès Alenia Space La société Thalès Alenia Space s'est constituée en 2007 par l'apport à Thalès des activités spatiales d'Alcatel.

Thalès Alenia Space est une référence mondiale dans le développement de technologies spatiales dans des domaines comme les télécommunications, la navigation, la météorologie, la gestion de l'environnement, la défense et l'observation.

L'entreprise est devenue en 2006 le leader mondial en terme de commandes et le premier constructeur en Europe dans le domaine des satellites.

En 2007, Thalès Alenia Space emploie 7200 personnes dans onze sites industriels dans quatre pays (France, Italie, Espagne, Belgique), et nous pouvons citer quelques grands programmes :

- les satellites de communication de la famille Spacebus dont Syracuse 3 (télécommunications militaires),
- les satellites en orbite basse de la famille Proteus (applications scientifiques avec le CNES),
- Galileo (navigation),
- Météosat seconde génération (météorologie),
- Helios (observation militaire)
- télescope spatial Herschel et satellite Planck (pour l'Agence spatiale européenne),...

2.3.1 Mission satellite

La figure I.5 représente une couverture typique d'une antenne dédiée multimédia par satellite, constituée ici de plusieurs dizaines de spots circulaires.

Cette couverture nécessite dans l'exemple 44 spots, pour deux raisons :

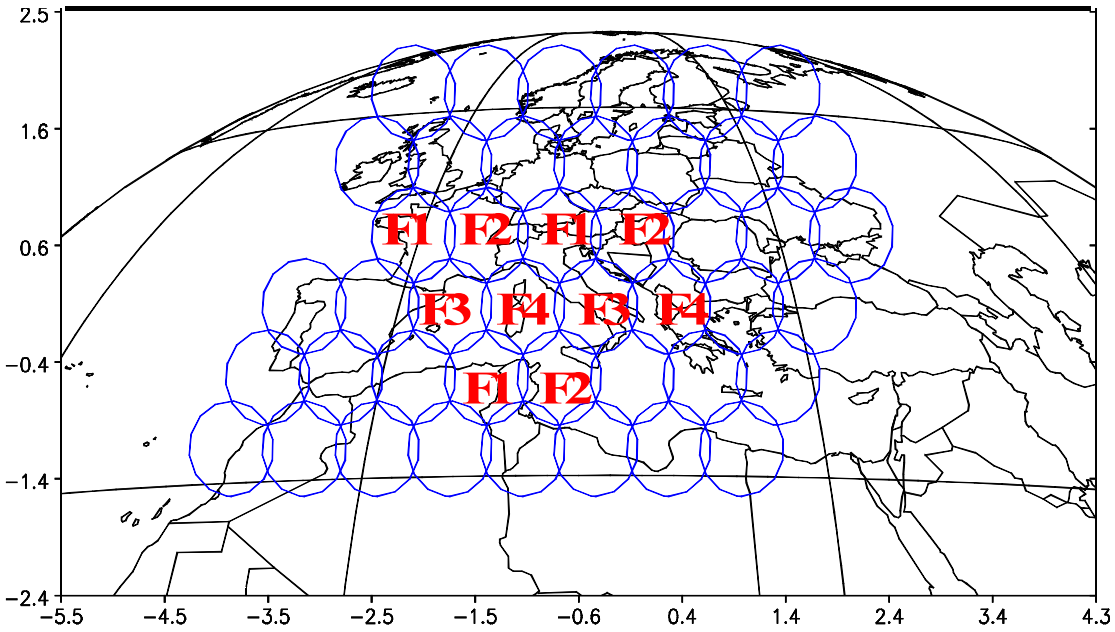


FIG. I.5 – Couverture multimédia de l'Europe par satellite

- Les bandes Ka (30/20 GHz pour les liaisons montante/descendante), réservées pour cette application pour leur large disponibilité fréquentielle, sont découpées chacune en 4 sous-bandes $F1, F2, F3, F4$, attribuées aux divers faisceaux selon un motif régulier (dans l'exemple un cercle).
  Cela permet de réutiliser $44/4 = 11$ fois chacune des sous-bandes et le système peut ainsi transmettre simultanément 11 fois plus de trafic.

- Le gain en directivité (inversement proportionnelle à la surface de son empreinte utile) de l'antenne augmente d'environ $10\ \log(44) > 16$ dB.
  Ceci permet d'assurer, malgré la forte atténuation atmosphérique en cas de pluie, une émission compatible avec de petites antennes usagers.

L'antenne doit former des faisceaux qui concentrent l'énergie du rayonnement émis alternativement sur chaque spot de la zone de couverture.

Nombre de sources Les antennes réseaux actives sont une solution intéressante pour les missions précédentes : l'utilisation d'un dispositif électronique appelé **répartiteur BFN** (pour *Beamforming Network*, c'est-à-dire réseau de formation de faisceaux) permet d'**appliquer la loi de phase adaptée pour générer le rayonnement désiré**.

Cette utilisation a été validée sur le satellite expérimental Stentor (entièrement mesuré au sol, même si l'échec au lancement d'Ariane 5 n'a pas permis de le faire en vol) conçu pour des applications moyen débit en bande dite Ku (10-12 GHz pour la télévision numérique par exemple).

Mais ce type d'antenne entraine des contraintes :

- Avec l'apparition de nouveaux systèmes géostationnaires multispots, l'antenne devient de plus en plus grande (environ 1 m de diamètre) afin d'assurer les contraintes de rayonnement requises avec des faisceaux fins.

- De plus, les lobes secondaires résultent de la périodicité du réseau (voir figure I.6), et la nécessité de les rejeter en dehors de la Terre dans notre cas implique de choisir une distance plus petite entre les sources.

Ces contraintes engendrent un très grand nombre (plusieurs centaines) de sources :

- ce nombre excède la dimension habituelle de recherche des lois d'alimentation (synthèse de réseau),

- et il faut construire autant de contrôles, qui sont des dispositifs électroniques coûteux.

Le coût d'une telle antenne devient excessif.

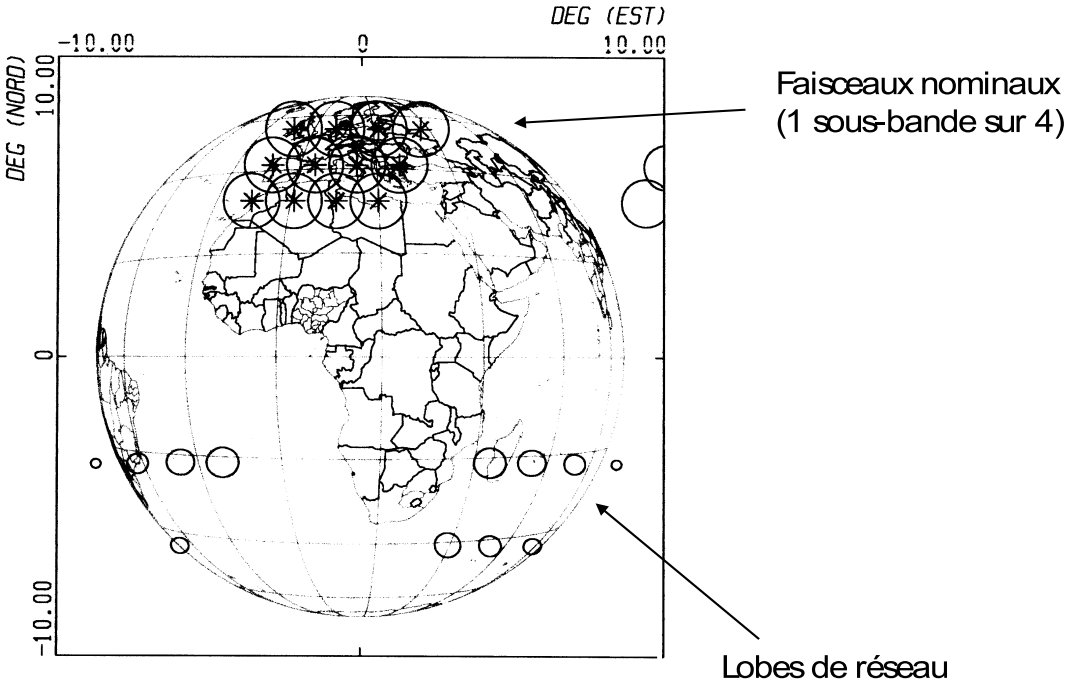


FIG. I.6 – Lobes de réseau

Objectif L'objectif de la collaboration est de minimiser le nombre de sources sans détériorer les performances de l'antenne : il faut optimiser le maillage du réseau qui deviendra irrégulier.

La méthode devra être automatisée au maximum afin d'aboutir à la création d'un logiciel.

2.3.2 Etudes Thalès précédentes

L'antenne de départ consiste en un réseau plan régulier d'ER, et lors de la minimisation du nombre d'ER, on veut briser la périodicité du réseau pour supprimer les lobes de réseau.

Nous présentons ici deux solutions testées par Thalès pour réduire le nombre de sources dans ce sens.

Antenne réseau raréfié La méthode consiste à désactiver certaines sources : on effectue des trous dans la grille de départ.

Cette variante, rapportée dans de vieilles publications sur les antennes radar, a été évaluée par Thalès.

Pour un niveau de lobes secondaires similaire, donc des performances équivalentes d'isolation entre faisceaux réutilisant la même sous-bande de fréquence, elle a permis de réduire le nombre d'ER de 253 à 189 (cela représente 75% du nombre d'ER de départ), selon la figure I.7, tout en augmentant un peu le diamètre de l'antenne.

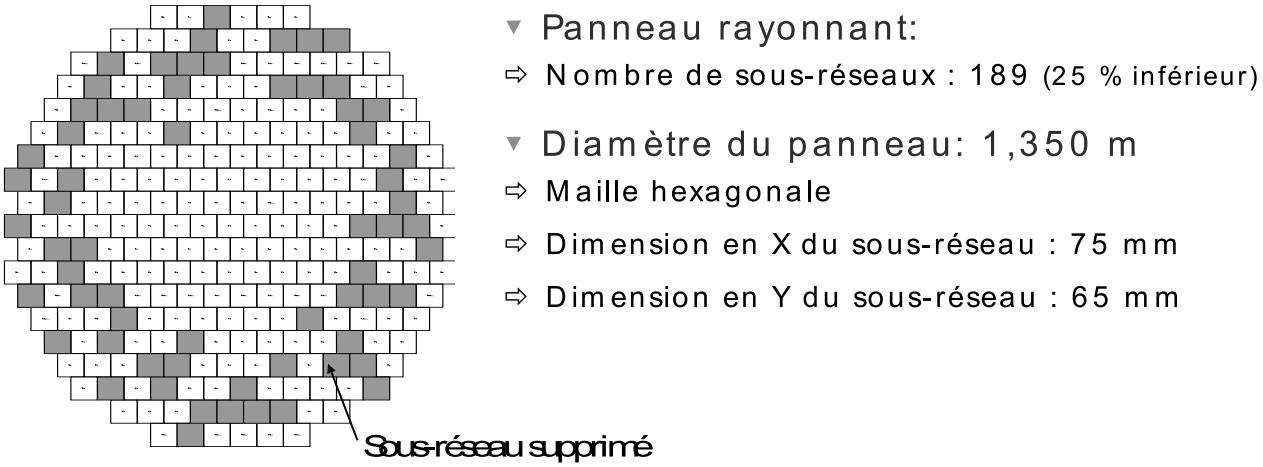


FIG. I.7 – Antenne réseau raréfié

Cette réduction est obtenue en optimisant la place des trous : à maille du réseau égale, les lobes de réseau sont légèrement abaissés, parce que l'antenne n'est plus purement périodique.

Comme autre avantage, la densité des trous augmente du centre vers la périphérie : ceci réalise une répartition d'énergie apodisée sur l'antenne (nécessaire pour réduire les niveaux des premiers lobes secondaires), tout en utilisant des amplificateurs à puissance identique derrière chaque ER. On maximise ainsi la puissance rayonnée, le rendement énergétique des amplificateurs, et on réduit le coût par effet de série.

Un autre test est présenté dans[Gui07] et [CCG+07] : pour un spot visé donné, une fonction coût est définie à partir des contraintes de rayonnement et minimisée avec deux algorithmes.

L'antenne de départ est composée d'un réseau régulier à maille triangulaire de 511 sources. Pour des performances similaires à l'antenne initiale, on obtient les résultats

suivants pour le spot n° 38, situé vers le centre de la zone de couverture (voir figure I.8) :

- en utilisant un algorithme génétique, le nombre final d'ER est 74% de celui de départ,
- en utilisant un algorithme de recuit-simulé, le nombre final d'ER est 58% de celui de départ.

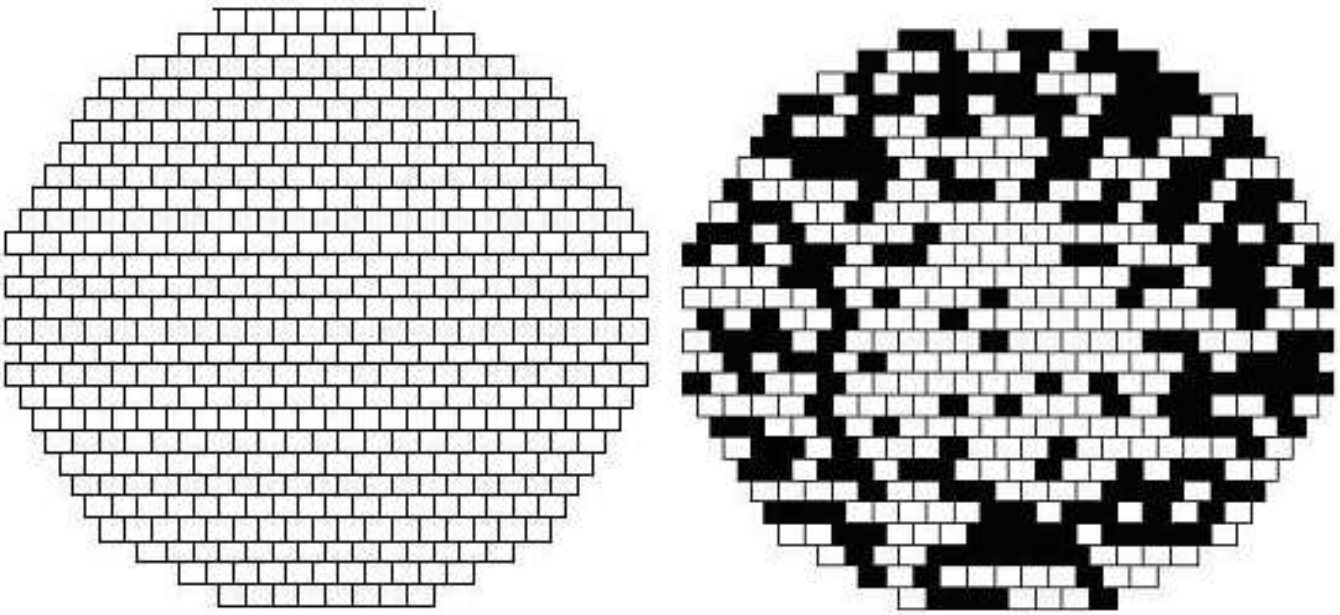


FIG. I.8 – Antenne réseau raréfiée
A gauche : antenne de départ à 511 ER (réseau régulier)
A droite : antenne optimale pour le spot n°38 (algorithme de recuit-simulé)

Mais cette solution présente l'inconvénient de réduire le gain total de l'antenne.

Antenne réseau avec regroupements d'ER Une deuxième solution a été évaluée avec le même réseau de départ : regrouper des sources voisines avec la même puissance d'alimentation.

La figure I.9 montre un exemple très simplifié, simulé par Thalès (études préparatoires pour le satellite Syracuse 3) : pour constituer un ER, les patches de base étaient regroupés par 8 dans la partie centrale, par 16 ailleurs.

Dans le test avec le réseau de 511 ER ([Gui07] et [CCG+07]), l'algorithme de recuit-simulé ayant donné de meilleurs résultats pour la méthode de réseau raréfié, il a été utilisé ici pour la minimisation de la fonction coût.

On obtient un nombre d'ER égal à 47% de celui de départ pour des performances similaires sur le spot n°38 (voir figure I.10).

Cette solution semble meilleure puisque celle-ci remplit entièrement la surface rayonnante (efficacité maximale au niveau du gain) et assure la répartition de puissance par une densité accrue d'ER au centre par rapport à l'extérieur.

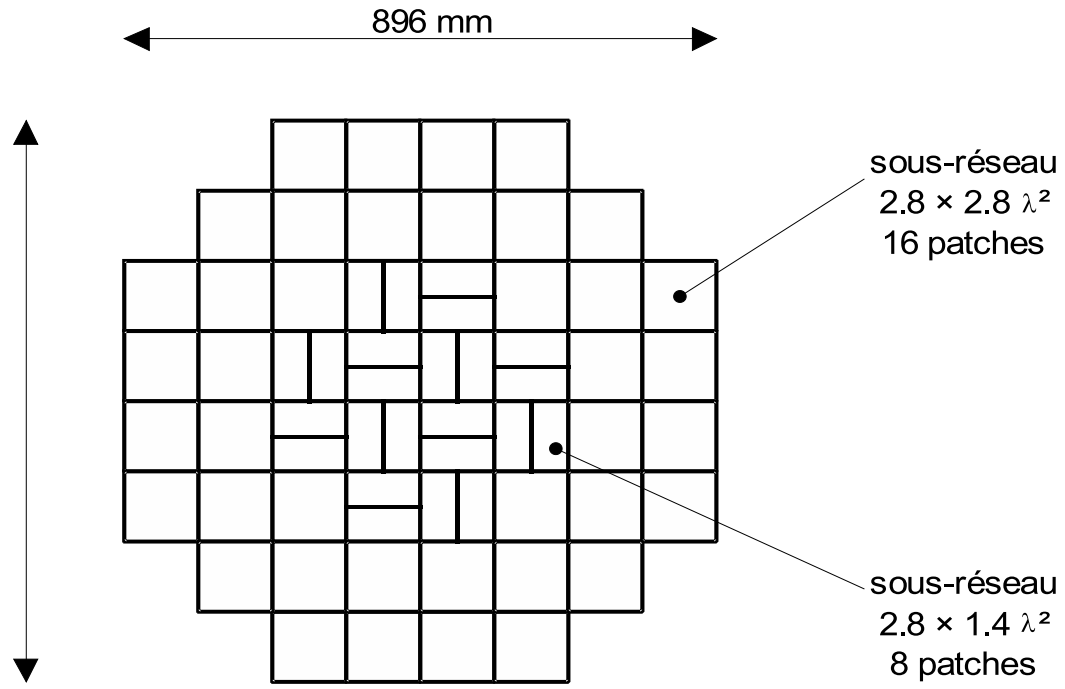


FIG. I.9 – Antenne de transmission : réseau à maille irrégulière

2.3.3 Objectif de l'étude

Les deux solutions précédentes permettent de réduire le nombre d'ER de manière conséquente, avec l'avantage au niveau du coût de fabrication d'utiliser un seul type d'amplificateur (dans la deuxième méthode, chaque regroupement d'ER est relié à un seul contrôle). Le regroupement d'ER permet d'obtenir une antenne finale avec moins de sources que le réseau raréfié.

Mais ces méthodes possèdent des défauts :

- L'optimisation du maillage du réseau est réalisée séparément pour chaque spot, et on ne sait pas réaliser une antenne optimale pour l'ensemble des spots.
- Dans la méthode de regroupement d'ER, on n'a pas tenu compte des contraintes industrielles de fabrication, et les géométries d'ER de l'antenne optimale ne sont pas toutes réalisables.

Le but de la collaboration entre Thalès et MIP est de mettre au point une méthode originale de conception : le point de départ est une antenne à réseau régulier et on veut utiliser la méthode du regroupement d'ER.

Il faudra s'affranchir des défauts précédents :

- On souhaite un algorithme qui définit une antenne optimale pour l'ensemble des spots, avec une méthode analytique qui trouve un minimum global dans un problème difficile où les nombreux minima locaux font échouer à priori les méthodes classiques de gradient.

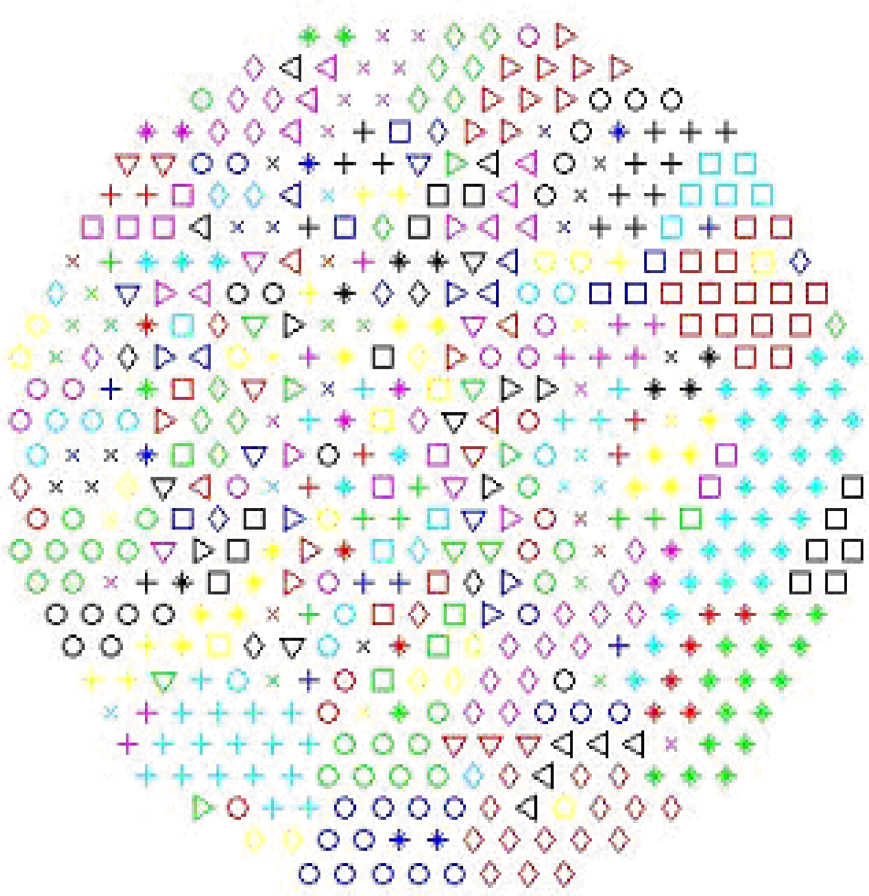


FIG. I.10 – Antenne réseau à maille irrégulière optimale pour le spot n°38
(les ER marqués avec le même symbole peuvent être regroupés)

- Les formes d'ER regroupés seront limitées, et il faudra augmenter le nombre de formes différentes d'ER, toujours de taille croissante du centre au bord, tout en résolvant les problèmes de géométrie associés (emboîtement des ER).

L'objectif est de concevoir un algorithme entièrement automatique.

2.3.4 Méthodes couramment utilisées

Synthèse de réseau Il existe de nombreuses formulations différentes d'un problème de synthèse d'une antenne réseau, et nous renvoyons le lecteur aux références [BEMP94], [Che71] et aux travaux de [Man89], [Roq98], [Fad05] pour une présentation plus détaillée de ces classes de méthodes.

Méthodes analytiques Dans cette catégorie, on déduit la loi d'alimentation par des formules.

Voici quelques exemples d'application :

- Dans [Che71] on trouve une synthèse en amplitude et phase pour une antenne réseau linéaire : la maximisation de la directivité est exprimée comme rapport de formes quadratiques.
- Dans dautres travaux, le diagramme est approché par une somme trigonométrique ou de polynômes de Chebishev (voir [Dol46] parmi les premiers travaux sur la synthèse de réseaux).
- Dans la méthode variationnelle, un développement analytique permet de déterminer une condition (sous forme de système carré d'équations non linéaires) pour qu'une variation des coefficients n'entraîne pas de variation du critère (voir [Man89]).

Beaucoup de ces méthodes ne s'appliquent qu'à des cas particuliers, par exemple :

- pour un type de gabarit ou de réseau précis,
- sans contraintes sur les paramètres.

Nous souhaitons disposer d'une méthode plus générale pour pouvoir l'appliquer à des antennes et des gabarits quelconques.

Méthodes d'optimisation globale Les problèmes de recherche d'une loi d'alimentation optimale peuvent se ramèner à la minimisation d'une fonction coût, définie comme l'erreur entre le diagramme de rayonnement et le gabarit, sous des contraintes techniques. Mais ce problème admet généralement de nombreux minima locaux, comme le soulignent les auteurs de [Cap06].

Des algorithmes d'optimisation globale, basés sur une exploration du domaine de définition (comme les algorithmes génétiques, ou de recuit-simulé), sont alors utilisés pour trouver la solution (voir une application dans [DA96]).

Dans cette catégorie, on trouve par exemple la méthode des perturbations : on part d'une alimentation initiale, et lors d'un processus itératif, on retient les perturbations de la loi d'alimentation qui apportent une diminution de la fonction erreur, jusqu'à ce qu'on ne puisse plus l'améliorer (voir [KM97]). On utilise ce type d'algorithme lorsqu'on ne dispose pas du gradient, sinon des méthodes plus efficaces existent.

Comme nous le verrons dans la section 2.3.3 du chapitre II, dans une synthèse de réseau, selon le type de norme choisie pour la fonction erreur, on doit résoudre un des problèmes d'optimisation suivants.

Problèmes de type minimax Le problème de type minimax admet diverses formulations, nous le notons :

$$\begin{cases} \min\limits_{x} \left(\max\limits_{i=1\ldots p} f_i(x) \right) := \min\limits_{x} \; J_1(x) \\ g_i(x) \leqslant 0, \quad i = 1 \ldots m \end{cases} \tag{I.1}$$

avec $f_i : \mathbb{R}^n \longrightarrow \mathbb{R}$.

Même si nous supposons les f_i différentiables, c'est un problème d'optimisation non différentiable à cause du critère "max".

On peut reformuler le problème et se ramener aux méthodes classiques d'optimisation différentiable. D'après [PM85], on peut donner les exemples suivants de synthèses de réseau :

- On remplace le gradient de J_1 par un gradient généralisé (par exemple le gradient généralisé de Clarke).
- On approche J_1 par une fonction différentiable (par exemple au sens de la norme L^p ou en linéarisant les f_i comme dans la méthode de Madsen, voir [SJM76]). Le principe est d'effectuer une succession d'approximations :
  - A l'itération k, on cherche le pas de descente h_k.
  - Pour cela on résoud un problème de programmation linéaire : $J_1^k(h_k) = \min_{\|h\| \leq \lambda_k} \left(J_1^k(h) \right)$ avec $J_1^k(h)$ une approximation linéaire de $J_1(x_k + h)$.
  - On construit ainsi la suite $x_{k+1} = x_k + h_k$ qui fait diminuer la fonction erreur et converge vers la solution.

Problèmes de type moindre carrés Le problème de type moindre carrés est un problème d'optimisation différentiable, on peut donc utiliser les méthodes classiques utilisant le calcul du gradient :

$$\begin{cases} \min\limits_x \left(\sum\limits_{i=1}^{p} f_i(x)^2 \right) := \min\limits_x \ J_2(x) \\ g_i(x) \leqslant 0, \quad i = 1 \dots m. \end{cases} \tag{I.2}$$

On trouve comme exemples d'applications à la synthèse de réseau :

- les méthodes duales utilisant les multiplicateurs de Lagrange dans [NV95],
- des méthodes de type projection utilisant la SVD (pour *Singular Value Decomposition*) comme dans [MP93].

Dans la plupart des cas, l'algorithme utilise des propriétés du diagramme de rayonnement liées à un type de réseau particulier.

Par exemple dans [BFT97], dans le cas d'un réseau plan régulier, la méthode utilise les propriétés quadratiques du diagramme pour mettre en place une stratégie de sortie des "puits" constitués par les minima locaux.

Géométrie de l'antenne Dans notre étude, nous voulons optimiser la géométrie de l'antenne suivant le critère de réduction du nombre d'ER, en faisant varier leur taille et leur forme.

La géométrie d'une antenne réseau dépend le plus souvent de contraintes matérielles de réalisation, et on trouve peu de travaux publiés concernant l'optimisation d'une géométrie.

Certains concernent cependant la recherche de la position des éléments rayonnants (ER) pour obtenir un diagramme fixé.

Ainsi, expliquée dans [Che71], la méthode des perturbations est utilisée dans le travail de [Lam93]

- Soit X_k le vecteur à l'itération k qui définit la position des ER sur l'antenne et $\sigma_k(X)$ l'erreur sur le vecteur G_k qui définit le gabarit,
- On part d'un système simple avec un gabarit de départ G_0 réalisable, c'est-à-dire qu'on sait calculer X_0 qui définit une géométrie des ER de départ donnant le minimum global de $\sigma_0(X)$,
- On provoque une suite de perturbations du gabarit notées $G_0, \ldots, G_q$: chaque perturbation doit être assez petite pour qu'en partant de la solution X_k du système au rang k, on obtienne la solution globale du système au rang $k+1$ avec l'algorithme d'optimisation choisi.
- On obtient une géométrie finale X_q correspondant au gabarit G_q qui est celui que l'on veut finalement satisfaire.

L'auteur souligne que des configurations de réseau très différentes peuvent toutes produire des diagrammes de rayonnement proches du gabarit, car dans tous les cas la valeur du minimum local trouvé est proche de celle du minimum global.

Il applique alors une phase de recherche globale aléatoire : on essaye de passer d'un minimum local à l'autre en estimant pour chacun par des coupes la dimension de la zone d'attraction et la valeur du minimum.

Dans ce cas, on choisit une fonction erreur qui présente le moins de minima locaux possibles, et la stratégie suivante est préconisée :

- Dans un premier temps, quitte à faire une approximation où on privilégie certaines directions, on déplace fortement les ER pour satisfaire un gabarit simplifié.
- Dans un deuxième temps, on déplace les ER relativement à la solution obtenue pour essayer de satisfaire le gabarit dans toutes les directions.

Choix d'une méthode Pour nous guider dans le choix d'une méthode, nous présentons dans la suite :

- les méthodes mathématiques d'optimisation globale classiques dans la section 3, car elles sont utilisées pour résoudre le problème de synthèse de réseau et même le problème de géométrie (voir par exemple [Gui07]),
- les méthodes mathématiques d'optimisation topologique dans la section 4.

3 Optimisation sous contraintes

Nous présentons ici les méthodes mathématiques d'optimisation sous contraintes classiques : pour les problèmes difficiles, qui possèdent de nombreux minima locaux, des méthodes globales sont généralement utilisées, mais en pratique on est rarement assuré de trouver le minimum global.

Un problème d'optimisation convexe possède un unique minimum global, et des méthodes locales performantes permettent de le trouver. Nous finissons la section avec l'**algorithme de Levenberg-Marquardt**, que nous utiliserons pour la recherche des lois d'alimentation optimales après avoir transformé le problème pour le rendre convexe.

3.1 Méthodes globales

3.1.1 Formulation

La recherche de minimum global sous contraintes est un problème difficile, et il n'existe pas d'algorithme pour le résoudre dans un cas quelconque.

De plus la complexité du problème augmente avec la taille de l'espace de recherche. Le coût de calcul de l'algorithme, lié au nombre d'évaluations de la fonctionnelle, et le nombre de minima locaux augmentent. Certaines méthodes ont ainsi de bonnes performances si on travaille avec quelques variables mais deviennent impraticables quand on passe à plusieurs dizaines.

Voici une formulation de ce type de problème : on dispose de paramètres x à optimiser dans un espace de conception Ω de dimension n, mais souvent la physique du problème impose des solutions admissibles dans un domaine Λ plus petit que Ω.

Cette information est introduite sous forme de m contraintes sur les paramètres à optimiser :

$$\Lambda = \{x \in \mathbb{R}^n \ ; \quad g_i(x) \leqslant 0, \ i = 1 \dots m\} \tag{I.3}$$

avec $g_i : \mathbb{R}^n \longrightarrow \mathbb{R}$.

On formule alors la minimisation de la fonction coût J (par exemple une fonction erreur) par rapport à la variable x sous les contraintes g_i :

$$\min_{x \in \Lambda} J(x) \qquad \Longleftrightarrow \qquad \left\{ \begin{array}{l} \min J(x) \\ g_i(x) \leqslant 0, \quad i = 1 \dots m \end{array} \right. \tag{I.4}$$

avec $J : \mathbb{R}^n \longrightarrow \mathbb{R}$.

Cependant, la difficulté du problème dépend en partie des propriétés topologiques de Λ, donc il n'est pas toujours pertinent d'augmenter la complexité de Λ en injectant toutes les informations disponibles sous forme de contraintes.

3.1.2 Problème d'optimisation convexe

Certaines hypothèses plus ou moins fortes sur J et Λ peuvent assurer l'existence d'un minimum global, mais il est difficile de caractériser la classe de problèmes d'optimisation sous contraintes qui ont un minimum global unique.

Le cas le plus favorable est celui d'une fonction coût J convexe sur un domaine de contraintes Λ convexe : tout minimum local est alors global, et la convergence des méthodes de descente est assurée (voir [HHL93]).

3.1.3 Classification

On peut présenter une classification générale des algorithmes d'optimisation globale avec le schéma de la figure I.11. Pour un exposé plus complet, le lecteur peut consulter [BGLS03], [Cia98], [Min83], [DPST03].

Suivant ce shéma, on distingue d'abord les problèmes d'optimisation :

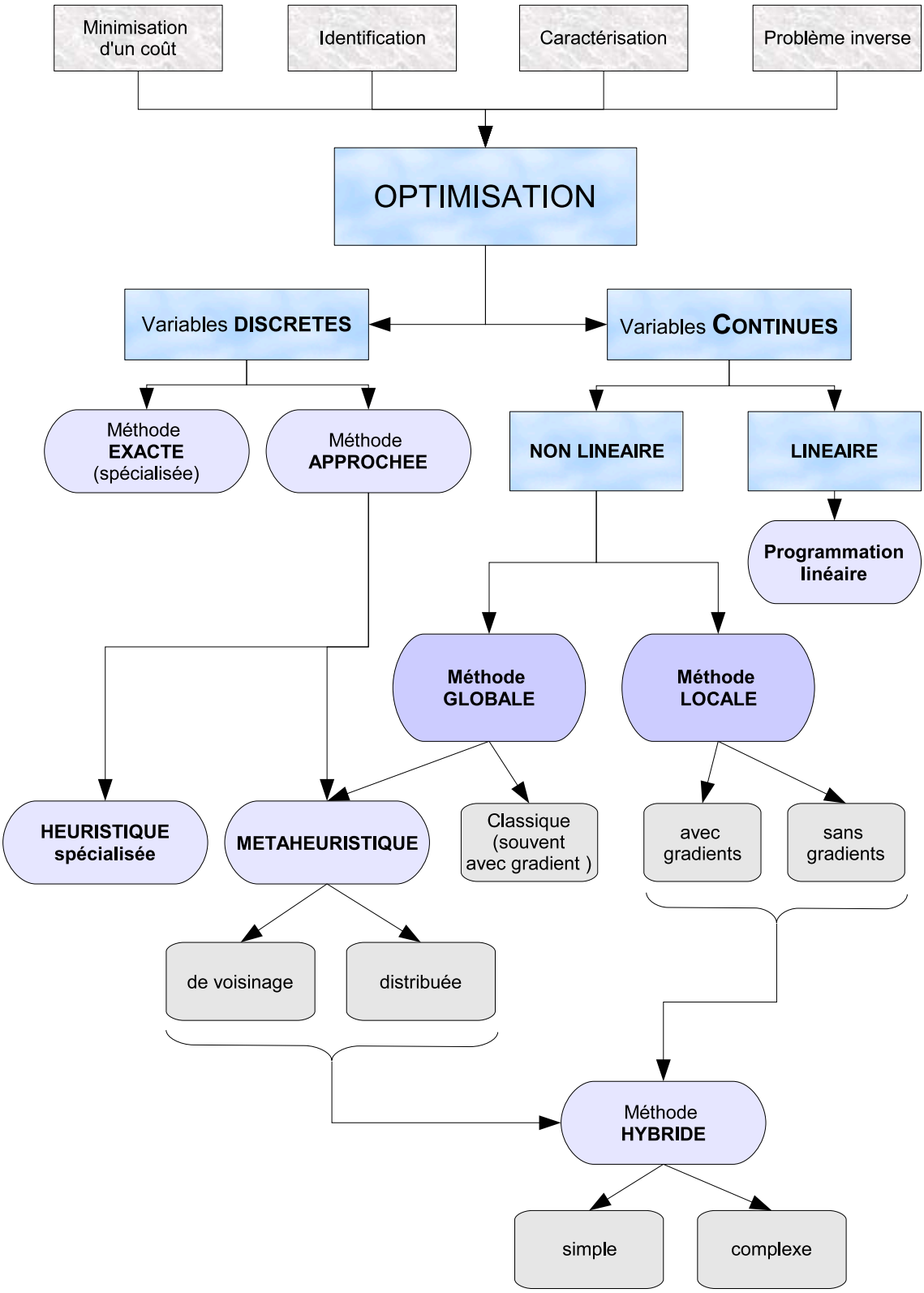


FIG. I.11 – Classification des méthodes d'optimisation (extrait de [DPST03])

- A variables discrètes avec comme exemples : le problème du voyageur de commerce, d'ordonnancement,... De nombreuses heuristiques ont été développpées, mais souvent elles ont été conçues pour pour un problème spécifique (heuristiques spécialisées).
- A variables continues avec comme exemples : la minimisation d'une fonction coût, le problème des valeurs à affecter aux paramètres d'un modèle numérique pour qu'il reproduise au mieux un comportement réel observé, les problèmes de moindres carrés...

Pour l'**optimisation continue**, qui concerne la synthèse de réseau,on distingue alors :

- L'optimisation linéaire : elle concerne une fonction coût linéaire sous contraintes linéaires, la programmation linéaire permet de résoudre le problème.
- L'**optimisation non linéaire** dans les autres cas :
  - On peut utiliser l'application répétée d'une **méthode locale**, qui est conçue pour trouver un minimum local d'un problème d'optimisation sans contrainte.

    On exploite ou non les gradients de la fonction coût, car quand celle-ci n'est pas connue analytiquement, le calcul des gradients est difficile.

    Cependant, les méthodes les plus performantes sont les **méthodes de descente**, utilisant le gradient (méthode de quasi-Newton, gradient conjugué,...) se trouvent dans cette catégorie : leur inconvénient est de converger vers le premier minimum local trouvé.
  - Si le nombre de minima locaux est très important, le recours à une méthode globale s'impose.

Remarque On rencontre aussi des problèmes mixtes, qui comportent à la fois des variables discrètes et continues, on adapte alors les méthodes décrites.

Parmi les **méthodes globales**, on trouve les catégories suivantes.

Méthodes classiques Ce sont des algorithmes déterministes qui exigent des **propriétés de régularité** pour la fonctionnelle et le domaine des contraintes.

Elles permettent alors de disposer de méthodes robustes avec l'assurance de trouver le minimum global (par exemple l'optimisation convexe).

Métaheuristiques Elles sont souvent utilisées quand le problème est trop compliqué.

Ces méthodes comportent une partie stochastique qui permet de faire face à l'explosion combinatoire des possibilités. Elles sont généralement d'origine discrète mais s'adaptent aux problèmes continus, et elles sont souvent directes (c'est-à-dire qu'elles ne recourent pas au calcul souvent problématique des gradients).

Elles sont inspirées par des analogies avec la physique ou la biologie, et partagent les mêmes inconvénients :

- difficultés de réglage des paramètres de la méthode,
- temps de calcul élevé (parfois prohibitif),
- manque de garantie d'obtenir la solution globale.

On trouve dans cette catégorie :

- les algorithmes génétiques ou évolutionnaires, colonies de fourmis,
- l'algorithme du recuit-simulé.

On peut encore différencier les métaheuristiques entre elles :

- "de voisinage", qui font progresser une seule solution à la fois (recuit simulé,...),
- "distribuées", qui manipulent en parallèle toute une population de solutions (algorithmes génétiques,...).

Méthodes hybrides Elles peuvent combiner plusieurs des algorithmes ci-dessus, s'efforçant de tirer parti des avantages spécifiques d'approches différentes.

Souvent elles associent une métaheuristique et une méthode locale.

Cette coopération peut prendre la forme d'un passage de relais entre la métaheuristique qui représente la phase globale d'exploration, et la méthode locale chargée dans une deuxième phase d'affiner la solution. On peut aussi avoir les deux approches entremêlées de manière plus complexe.

3.1.4 Approches globales

Les méthodes d'optimisation globale qui utilisent des stratégies pour se dégager du piège des minima locaux se déroulent souvent en deux phases :

- phase locale : recherche d'un point qui améliore la fonction coût,
- phase globale : recherche d'une meilleure estimation de la solution globale et d'un nouveau point de départ pour la phase locale.

  On peut par exemple autoriser de temps en temps des mouvements de remontée, c'est-à-dire une dégradation temporaire de la situation, avec un mécanisme de contrôle qui permet d'éviter la divergence du procédé. Il devient alors possible de s'extraire du "puits" constitué par un minimum local pour repartir explorer une autre zone.

On trouve dans cette catégorie les métaheuristiques et les méthodes hybrides évoquées précédemment.

Stratégies mixtes On peut utiliser plusieurs approches précédentes conjointement.

Par exemple, on peut ici aussi travailler en deux phases :

- phase globale par méthode d'approximation : on applique une méthode globale à un modèle simplifié, qui donne un premier point solution,
- phase locale : à partir du point trouvé, on applique une méthode locale au modèle non simplifié.

Suivant la configuration, ou le point de départ choisi, l'algorithme converge vers un minimum local qui est la meilleure des solutions accessibles compte tenu de l'hypotèse initiale, qui n'est généralement pas la solution globale espérée. Dans la pratique, on se contente d'améliorer une solution existante.

On peut appliquer l'algorithme de recherche locale plusieurs fois avec des conditions initiales différentes pour explorer les solutions obtenues, mais souvent le nombre de minima locaux augmente de façon exponentielle avec le nombre de paramètres à optimiser, et cette stratégie devient inefficace.

3.1.5 Stratégies d'approximation

On peut considérer qu'il y a deux approches basées sur des approximations pour résoudre numériquement un problème d'optimisation.

Reformulation du problème Une reformulation du problème, quitte à simplifier le modèle, peut nous permettre de travailler avec une fonction coût et des contraintes régulières, et ainsi d'utiliser une méthode classique avec les hypothèses adéquates.

Par exemple on peut rendre le problème convexe : alors dans ce cas précis tout minimum local est global, et les algorithmes d'optimisation de type descente convergent vers la solution.

Bien que théoriquement tout problème d'optimisation non convexe puisse se ramener à un problème convexe en considérant l'enveloppe convexe, cette transformation est souvent difficile (et même si la fonction est convexe mais si Λ n'est pas convexe, le problème peut être difficile), voir [HHL93] pour plus de détails.

De plus, on ne peut pas toujours effectuer des simplifications sur le modèle car la fonction coût peut être trop compliquée ou bien parce qu'on dégrade trop la solution.

Approximation du modèle Si la fonction de modélisation ne fournit pas le gradient, ou est fortement non linéaire, on peut utiliser à sa place une approximation régulière pour laquelle le gradient est disponible et appliquer ensuite une méthode de descente, plus performante que les méthodes sans gradient.

Pour approcher la fonction de modélisation, on peut recourir aux méthodes de surface de réponse ou d'interpolation (réseaux de neurones, splines, *sparse grids*,...) qui s'adaptent le mieux au problème. Dans cette catégorie, on peut citer le travail de [VG04] au laboratoire MIP et de [Fad05] concernant la synthèse de réseau.

Pour des problèmes difficiles d'optimisation globale, on peut combiner les deux stratégies précédentes.

3.1.6 Conclusion

Actuellement, il est difficile de prévoir l'efficacité d'une de ces méthodes donnée, appliquée à un problème donné.

La plupart des heuristiques utilisent des paramètres : le choix de ces paramètres est simple et donne de bons résultats avec des problèmes tests, ou bien la théorie peut préconiser des valeurs de réglage optimal, mais l'application à une fonctionnelle issue d'un cas réel est souvent difficile.

Ainsi, lors de l'implémentation de ces algorithmes sur des cas industriels, le réglage de la méthode fait souvent appel à l'expérience de l'utilisateur.

3.2 Idée clé : reformulation en un problème convexe

3.2.1 Choix d'une méthode d'optimisation

Critères de sélection Face à un problème d'optimisation concret, la principale difficulté est le choix d'une méthode capable de produire une solution acceptable au prix d'un temps de calcul raisonnable. La théorie n'est pas toujours d'un grand secours car on ne dispose pas généralement de théorèmes de convergence.

Ainsi, comme pour le réglage des paramètres d'une heuristique, le choix d'une bonne méthode d'optimisation fait appel au savoir faire de l'utilisateur.

La détermination de la méthode de synthèse de réseau repose donc sur le choix :

- de la fonction d'erreur,
- des contraintes sur les paramètres,
- de l'algorithme d'optimisation en fonction de critères de performance :
  - stabilité (faible sensibilité aux erreurs et aux variations des conditions initiales),
  - temps de calcul (comportement de la méthode quand le nombre de sources est grand, ici plusieurs centaines).

Idée clé : reformuler le problème pour le rendre convexe En ce qui concerne la synthèse de réseau, dans notre étude, la directivité qui est donnée sous forme analytique, donc nous disposons facilement du gradient de la fonction coût, et nous avons décidé d'utiliser une méthode de type gradient pour bénéficier de son efficacité : l'inconvénient est qu'elle trouvera le minimum local le plus proche du point de départ, et pas forcément le minimum global qui est celui recherché.

L'idée clé est alors de transformer le problème en un **problème convexe numériquement équivalent**.

On remplace alors un problème difficile (synthèse d'un réseau de plusieurs centaines de sources) avec un grand nombre de minima locaux par un problème convexe : le **minimum global unique du problème transformé** peut être trouvé en appliquant une méthode de descente, et il est la **solution du problème initial**.

Comme il n'y a pas de méthode générale pour tranformer le problème, le lecteur se reportera aux sections présentant les tests numériques pour savoir dans chaque cas comment nous avons procédé.

En ce qui concerne la méthode de descente, nous avons choisi l'algorithme de Levenberg-Marquardt, qui est une méthode de quasi-Newton appliquée à un problème de moindres carrés, pour le bon compromis entre efficacité et coût de calcul qu'elle nous semble présenter dans notre cas.

3.3 Méthodes de descente

Nous détaillons dans la suite les méthodes d'optimisation de type descente, jusqu'au rappel de l'algorithme de Levenberg-Marquardt.

3.3.1 Introduction

On veut résoudre le problème de minimisation sans contrainte avec une méthode itérative :

$$(\mathcal{P}_m) \quad \min_x J(x) \tag{I.5}$$

avec $J : \mathbb{R}^n \longrightarrow \mathbb{R}$ que l'on suppose suffisamment régulière.

Direction et pas de descente Une direction de descente $d_k \in \mathbb{R}^n$ est caractérisée par

$$\exists t \in \mathbb{R}^{+*} \quad J(x_k + t\ d_k) < J(x_k)$$

ou encore

$$d_k^T\ \nabla J(x_k) < 0.$$

Généralement on calcule une direction de descente d_k et on cherche le pas sous la forme

$$p_k = t_k\ d_k \quad , t_k \in \mathbb{R}^{+*}$$

de façon à assurer de diminuer localement la valeur de J avec $J(x_k + p_k) < J(x_k)$.

Cela permet d'obtenir une suite de points qui converge vers le minimum local le plus proche de x_0 avec

$$x_{k+1} = x_k + p_k$$

.

Recherche linéaire Une recherche linéaire consiste à calculer t_k en minimisant la fonction réelle q suivante :

$$t \longrightarrow q(t) = J(x_k + t\ d_k). \tag{I.6}$$

Cela revient à minimiser J dans la direction de descente d_k pour trouver le pas de descente $p_k = t_k\ d_k$.

Remarque : q est une fonction décroissante au voisinage de 0 ($q'(t) = d_k^T\ \nabla J(x_k + t\ d_k)$ donc $q'(0) = d_k^T\ \nabla J(X_k) < 0$ car d_k est une direction de descente).

Principe L'algorithme d'une méthode de descente est alors :

- initialisation : $x_k = x_0 \in \mathbb{R}^n$
- tant que le test d'arrêt est non vérifié
  - calculer une direction de descente $d_k \in \mathbb{R}^n$
  - calculer un pas de descente $p_k = t_k\ d_k$, $t_k \in \mathbb{R}^{+*}$

3.3.2 Méthodes classiques

Algorithme de la plus forte pente La méthode la plus élémentaire consiste à choisir dans l'algorithme précédent comme direction de descente l'opposé du gradient

$$d_k = -\frac{\nabla J(x_k)}{\|\nabla J(x_k)\|}$$

.

Algorithme du gradient conjugué La méthode du gradient conjugué s'applique à une fonction quadratique, et ses caractéristiques sont :

- A chaque étape k la direction d_k est obtenue comme combinaison linéaire du gradient courant et de la direction précédente $d_k = -\nabla J(x_k) + \beta_k\ d_{k-1}$.
- Le calcul du pas optimal p_k est exact.
- Les coefficients β_k sont choisis de telle manière que d_k soit conjuguée avec toutes les directions précédentes, ce qui permet la convergence en n itérations.

La méthode de Fletcher-Reeves est une extension du gradient conjugué pour une fonction quelconque, les différences sont :

- Une autre formule pour le calcul des coefficients β_k qui permettent d'obtenir la direction d_k.
- Le calcul du pas p_k s'effectue par recherche linéaire.
- Un test d'arrêt est nécessaire.

Cette méthode est intéressante car elle ne nécéssite pas de stocker une matrice et sa vitesse de convergence est très supérieure à celle de la méthode du gradient.

Méthode de Newton Nous détaillons ici un algorithme fondamental parmi les méthodes de descente.

Soit $x_k \in \mathbb{R}^n$ le point courant, on va remplacer J par son approximation quadratique $\tilde{J}$, approximation valable au voisinage de x_k, en considérant le développement de Taylor à l'ordre 2 :

$$\tilde{J}(h) := J(x_k) + h^T\ \nabla J(x_k) + \frac{1}{2}\ h^T\ \nabla^2 J(x_k)\ h\ \simeq J(x_k + h) \tag{I.7}$$

avec

- $\nabla J(x_k) \in \mathbb{R}^{n\times 1}$ le gradient de J en x_k,

- $\nabla^2 J(x_k) \in \mathbb{R}^{n\times n}$ la hessienne de J en x_k.

Pour trouver x_{k+1}, on calcule le minimum de l'approximation $\tilde{J}(h)$:

- Si $\nabla^2 J(x_k)$ est définie positive, alors ce minimum existe et est unique.
- Il est donné par le point critique solution de :

$$\nabla \tilde{J}(h) = \nabla J(x_k) + \nabla^2 J(x_k)\ h = 0 \Leftrightarrow \nabla^2 J(x_k)\ h = -\ \nabla J(x_k). \tag{I.8}$$

On résoud le système linéaire précédent et la solution h_k est la direction de Newton, qui est une direction de descente si $\nabla^2 J(x_k)$ définie positive.

L'intérêt de cette méthode est que la suite des itérés converge de manière quadratique, mais uniquement si le point de départ est proche de la solution, ce qui en limite l'intérêt. Il faut d'autre part calculer la hessienne, ce qui est souvent prohibitif en terme de coût de calcul.

3.3.3 Algorithme de Levenberg-Marquardt

Méthodes de quasi-Newton Le principe d'une méthode de quasi-Newton (voir par exemple [BGLS03]) est de remplacer $\nabla^2 J(x_k)$ par une approximation moins coûteuse G_k.

On obtient une direction d_k en résolvant le système linéaire

$$G_k\ d_k = -\ \nabla J(x_k). \tag{I.9}$$

La condition pour assurer une direction de descente est : G_k matrice définie positive (en effet, on a alors $d_k^T\ \nabla J(X_k) = -\ d_k^T\ G_k\ d_k < 0$).

Suivant la stratégie de construction de G_k on obtient des algorithmes dont les plus connus sont l'algorithme DFP (pour Davidon, Fletcher, Powel) et l'algorithme BFGS (pour Broyden, Fletcher, Goldfarb et Shanno).

Méthode de Gauss-Newton On considère ici $f(x) := (f_1(x) \dots f_p(x))^T$ avec $f_j : \mathbb{R}^n \to \mathbb{R}$ une fonction erreur par exemple, et on veut résoudre le système d'équations non linéaires $f(x) = 0$ au sens des moindres carrés.

On construit à cet effet une fonction coût à minimiser

$$J(x) := \frac{1}{2}\ \|f(x)\|_2^2 = \frac{1}{2}\ f(x)^T\ f(x) = \frac{1}{2}\ \sum_{i=1}^{p} f_i(x)^2. \tag{I.10}$$

La méthode de Gauss-Newton est une méthode de quasi-Newton appliquée à la minimisation de J.

Le gradient et la hessienne prennent ici des formes particulières :

$$\begin{aligned} \nabla J(x_k) &= Df^T(x_k)\ f(x_k) \\ &= \sum_{j=1}^{p} f_j(x_k)\ \nabla f_j(x_k) \\ \nabla^2 J(x_k) &= Df^T(X_k)\ Df(X_k) + \sum_{j=1}^{p} f_j(x_k)\ \nabla^2 f_j(x_k). \end{aligned} \tag{I.11}$$

Dans le calcul de $\nabla^2 J(x_k)$, à proximité de la solution les f_i sont petits, et en négligeant alors le deuxième terme on définit la matrice de Gauss-Newton

$$G_k := Df^T(x_k)\ Df(x_k) = \sum_{j=1}^{p} \nabla f_j(x_k)\ \nabla f_j(x_k)^T \simeq \nabla^2 J(x_k). \tag{I.12}$$

Cette matrice G_k possède une propriété intéressante : elle est symétrique semi-définie positive.

La vitesse de convergence au voisinage d'un minimum de cette méthode dépend de combien le terme $Df^T(x_k)\ Df(x_k)$ domine le deuxième terme négligé de la hessienne $\nabla^2 J(x_k)$ (voir [NW99]) : dans les cas favorables, elle est de l'ordre de celle de la méthode de Newton (quadratique), pour un coût de calcul très inférieur .

Méthode de Levenberg-Marquardt On modifie la méthode précédente : on doit résoudre le système linéaire $G_k\ d_k = -\ \nabla J(x_k)$.

On pose

$$\widetilde{G_k} = G_k + \rho\ I_n \tag{I.13}$$

avec
- $\rho \in \mathbb{R}^{+*}$,
- I_n la matrice identité.

L'idée de l'algorithme de Levenberg-Marquardt (initialement [Lev44] et [Mar63]) est d'apporter une correction à G_k qui ne perturbe pas la solution et qui permet :

- de rendre G_k aussi proche que possible d'une matrice définie positive, de manière à assurer une direction de descente et accélérer la convergence,
- de conserver la propriété de symétrie de G_k.

Pour résoudre le système linéaire, on peut utiliser :

- si le rang $\widetilde{G_k}$ est maximal, une méthode classique pour inverser $\widetilde{G_k}$,
- sinon, une méthode de gradient conjugué préconditionné.

Finalement, le système linéaire à résoudre pour trouver la direction de descente d_k au rang k est

$$\left(\sum_{j=1}^{p} \nabla f_j(x_k)\ \nabla f_j(x_k)^T\ +\ \rho\ I_n \right)\ d_k = -\ \sum_{j=1}^{p} f_j(x_k)\ \nabla f_j(x_k). \tag{I.14}$$

La vitesse de convergence locale de cette méthode est similaire à celle de Gauss-Newton (voir [Kel99]). En pratique, dans de nombreux cas la convergence est rapide, ce qui fait de l'algorithme de Levenberg-Marquardt une méthode efficace avec un coût de calcul réduit.

4 Optimisation topologique

Dans notre étude, nous voulons optimiser la géométrie de l'antenne en minimisant le nombre d'ER.

Nous présentons dans la suite les méthodes de l'optimisation topologique, présentées dans [BS03] : nous finissons par les méthodes utilisant la notion de **gradient topologique**, car nous utiliserons une version simplifiée de ce concept dans l'algorithme de conception de l'antenne.

4.1 Formulation

L'industrie (aéorospatiale, automobile,...) utilise l'optimisation topologique pour la conception des structures mécaniques ou éléments électromagnétiques (par exemple [Cal04]).

Un problème d'optimisation de forme d'un objet est défini avec :

- Des paramètres de conception $x = (x_1, \ldots, x_n)$ soumis à d'éventuelles contraintes $g_i(x) \leq 0$, $i = 1 \ldots m$.

  Ces paramètres définissent ici une forme ω, ouvert borné dans un domaine de référence $\Omega \in \mathbb{R}^d$, $d = 2, 3$ de l'espace. On note dans la suite ω pour x.

- Un modèle dont la solution u permet d'évaluer le comportement (mécanique, électromagnétique,...) de l'objet.

  Le cas le plus courant consiste à résoudre une équation d'état $E(u(\omega), \omega) = 0$ avec une méthode d'éléments finis.

- Un critère ou fonction coût que l'on cherche à minimiser

$$J : \quad \mathbb{R}^n \quad \rightarrow \quad \mathbb{R}. \tag{I.15}$$

La formulation mathématique la plus générale du problème d'optimisation topologique est alors

$$\begin{cases} \min\limits_{\omega \in \Omega} J(u(\omega), \omega) \\ g_i(u(\omega), \omega) \leq 0, \quad i = 1 \ldots m. \end{cases} \tag{I.16}$$

Dans la suite on note de manière plus simple

$$\begin{cases} \min\limits_{\omega \in \Omega} J(\omega) \\ g_i(\omega) \leq 0, \quad i = 1 \ldots m. \end{cases} \tag{I.17}$$

Remarques

- En pratique, les fonctions J et g_i sont la masse, ou le déplacement, ou la compliance qui mesure la rigidité,...
- La topologie de l'objet en dimension deux est caractérisée par le nombre de "trous".

4.1.1 Méthodes

Selon la nature des paramètres employés et par ordre croissant de complexité, on distingue les méthodes suivantes, avec une illustration sur la figure I.12.

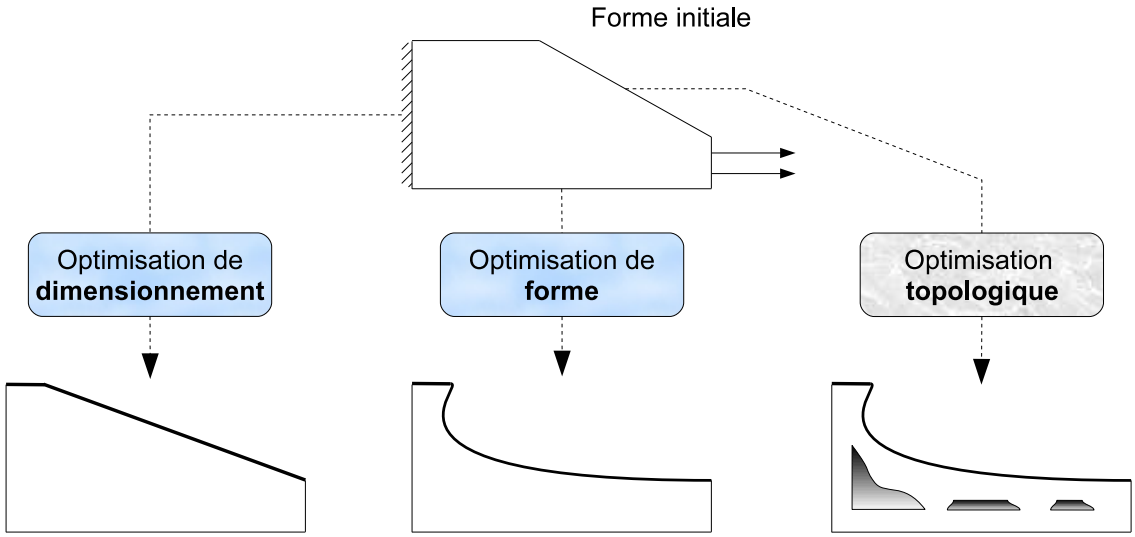


FIG. I.12 – Les méthodes de l'optimisation topologique

Optimisation de dimensionnement Les paramètres sont les longueurs, les sections, les épaisseurs,... Les parties fixes sont la géométrie et la topologie.

Avec l'optimisation de dimensionnement , on limite considérablement la variété des formes possibles.

Optimisation de forme (ou géométrique) Les paramètres sont les contours ou les surfaces (c'est-à-dire la géométrie). La partie fixe est la topologie.

La limitation vient ici du fait que le résultat dépend fortement de la forme initiale : on part à priori d'une forme connue, on obtient des formes successives au cours des itérations en faisant varier continuement les frontières de l'objet.

C'est la méthode employée par les logiciels d'optimisation classiques dans les bureaux d'études, qui ont remplacé la conception par essais successifs. Elle s'applique à des modèles et fonctions coût quelconques, mais elle peut être coûteuse à cause du remaillage si le modèle est une équation résolue numériquement On obtient souvent de nombreux minima locaux, et la forme finale reste alors dans des schémas classiques.

Optimisation topologique Le paramètre est la topologie, il n'y a pas ici de restrictions sur la forme.

On s'autorise ici à créer ou faire disparaître de la matière (trous, changements de connexion,...), cela permet en phase de préconception de remettre en question la forme habituelle.

Dans la pratique, si la forme trouvée ne répond pas aux contraintes techniques ou physiques, on peut ajouter une phase optimisation de forme après l'optimisation topologique, dont le rôle est alors de donner les caractéristiques générales de la structure.

Mise en oeuvre Les algorithmes en optimisation de forme et topologique consistent à partir d'une forme de départ w_0, et par itérations successives, trouver des formes w_k qui améliorent la fonction coût J jusqu'à la forme optimale.

Pour l'initialisation, on peut prendre comme forme initiale :

- En optimisation de forme : une forme connue (une solution existante).
- En optimisation topologique :
  - Une forme pleine, par exemple tout le domaine Ω : on va enlever de la matière lors des itérations.
  - Une forme vide : on va ajouter de la matière.
  - Une forme intermédiaire : on peut fixer certaines parties du domaine Ω comme vides ou pleines, cela revient à restreindre l'ensemble des formes admissibles.

Dans les sections suivantes, nous présentons les grandes méthodes d'optimisation topologique.

4.2 Méthodes de distribution de matière

On trouvera un exposé sur les méthodes suivantes dans [Ben89].

4.2.1 Paramétrisation de la forme

Le problème d'optimisation topologique consiste à répartir un matériau dans le domaine Ω pour créer une forme.

La forme $\omega(x, \rho(x))$ est paramétrisée ici par un point x du domaine et la densité de matière ρ en ce point :

- une forme classique par une densité de matière discrète, c'est-à-dire une fonction caractéristique notée χ

$$\begin{array}{rccll} \chi : & \Omega & \to & \{0,1\} & \\ & x & \mapsto & 0 & \text{si } x \notin \omega \text{ absence de matériau (trou)} \\ & & \mapsto & 1 & \text{si } x \in \omega \text{ présence de matériau} \end{array}$$

- une forme généralisée par une densité de matière continue ρ

$$\begin{array}{rccll} \rho : & \Omega & \to & [0,1] & \\ & x & \mapsto & 0 & \text{si } x \notin \omega \\ & & \mapsto & a > 0 & \text{si } x \in \omega \end{array}$$

Ici, la présence de matériau avec une densité intermédiaire doit être interprétée par la méthode (par exemple comme un matériau composite).

4.2.2 Discrétisation

Le problème (I.17) est continu en espace (paramètre x), pour le résoudre numériquement, une idée naturelle est de discrétiser.

On crée un maillage de conception, qui peut être différent d'un éventuel maillage éléments finis, et on distribue le matériau sur chaque élément de ce mailage. On obtient un problème d'optimisation discrète.

Limitations

- Si le maillage de conception n'est pas grossier, c'est un problème de grande taille, or comme l'optimisation discrète est coûteuse en calcul, on ne connaît pas d'algorithme efficace.
- C'est un problème mal posé, la solution présente une dépendance au maillage :

  si on résoud une suite de problèmes avec des maillages de conception de plus en plus fins, la suite des solutions ne converge pas. Quand on raffine le maillage, on obtient des formes avec de plus en plus de trous de petite taille au lieu d'obtenir une représentation plus précise d'une même structure optimale.

4.2.3 Relaxation : méthodes d'homogénéisation et SIMP

Une autre stratégie consiste à travailler avec une forme généralisée, définie dans la section 4.2.1 avec une densité continue entre 0 (pas de matière) et 1 (de la matière) : on autorise une infinité de matériaux, on dit qu'on a relaxé le problème.

En élargissant l'espace des formes admissibles, la relaxation rend les problèmes bien posés.

D'un point de vue mécanique, on considère un matériau dont il est nécessaire de définir les propriétés macroscopiques notées A_ρ pour les densités intermédiaires entre 0 et 1, qui interviennent dans les calculs.

On peut utiliser pour cela les méthodes :

- d'homogénéisation (voir [BK88] , [All01]) : le matériau est défini comme composite avec une microstructure particulière qui permet de déduire les propriétés macroscopiques et donc de fixer A_ρ.

  On obtient la solution comme une forme généralisée constituée de densités intermédiaires, non fabricable. Pour obtenir une forme non composite constituée de plein ($\rho = 1$) et de vide ($\rho = 0$), on applique une étape de pénalisation :
  on effectue quelques itérations supplémentaires où on force la densité à prendre des valeurs proches de 0 et 1.
- SIMP (pour *Solid Isotropic Material with Penalization*, voir [Ben89]) : on utilise un matériau fictif en fixant arbitrairement A_ρ. Elle est plus simple car on n'a plus besoin de définir de microstructure associée.

  Pour la large classe de problèmes consistant à minimiser la masse en maximisant la rigidité, l'étape de pénalisation n'est pas nécessaire car elle est implicite :

dans la formulation de la méthode, les grandes valeurs d'un paramètre p amènent les densités intermédiaires à prendre naturellement lors de l'optimisation les valeurs 0 ou 1.
Cependant, si on choisit une valeur de p trop grande, on obtient des minima locaux et donc une solution sensible au choix de la forme de départ. Dans la pratique, on applique plusieurs fois l'algorithme en augmentant lentement la valeur de p.

On a alors les caractéristiques :

- avantage : on peut utiliser des algorithmes d'optimisation continue efficaces,
- inconvénient : méthode limitée pour l'instant au cas de l'élasticité linéaire.

Les deux schémas des figures I.13 et I.14 illustrent le principe des méthodes précédentes.

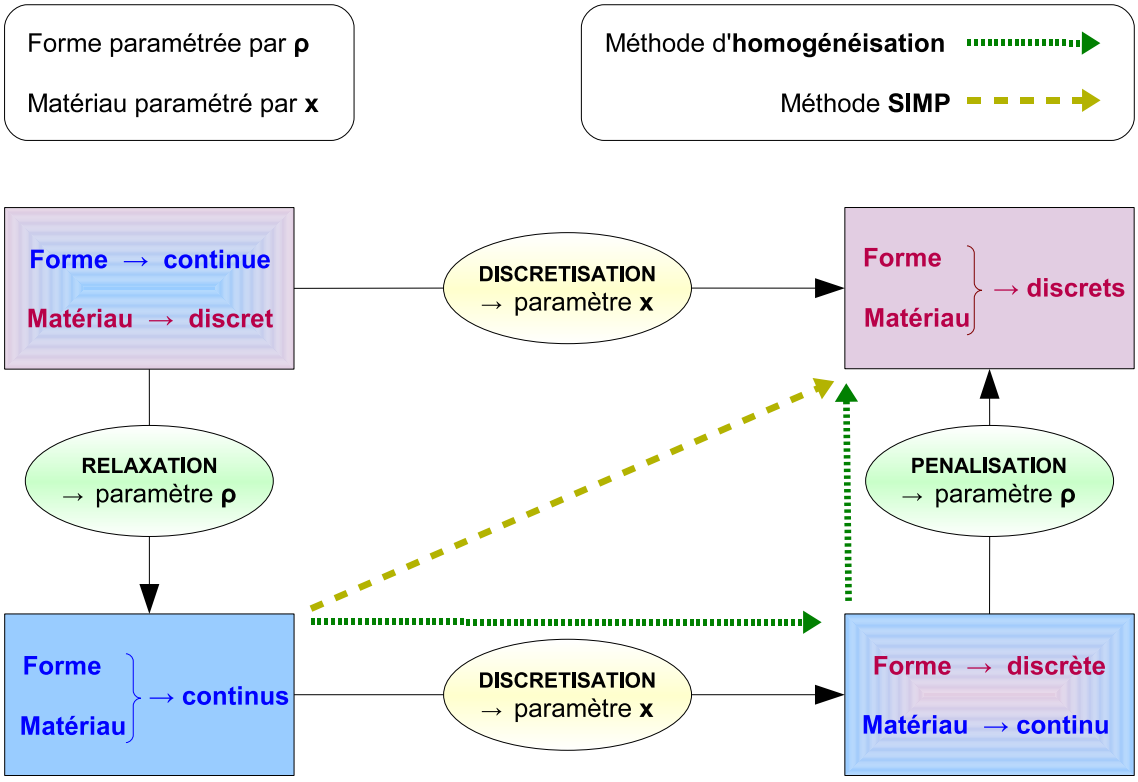


FIG. I.13 – Méthodes de relaxation : principe

4.3 Méthodes utilisant un gradient de forme

Pour un exposé des méthodes de cette section, le lecteur peut se référer à [SZ92].

4.3.1 Paramétrisation de la forme

La forme ω est paramétrisée ici par sa frontière notée $\delta\omega$.

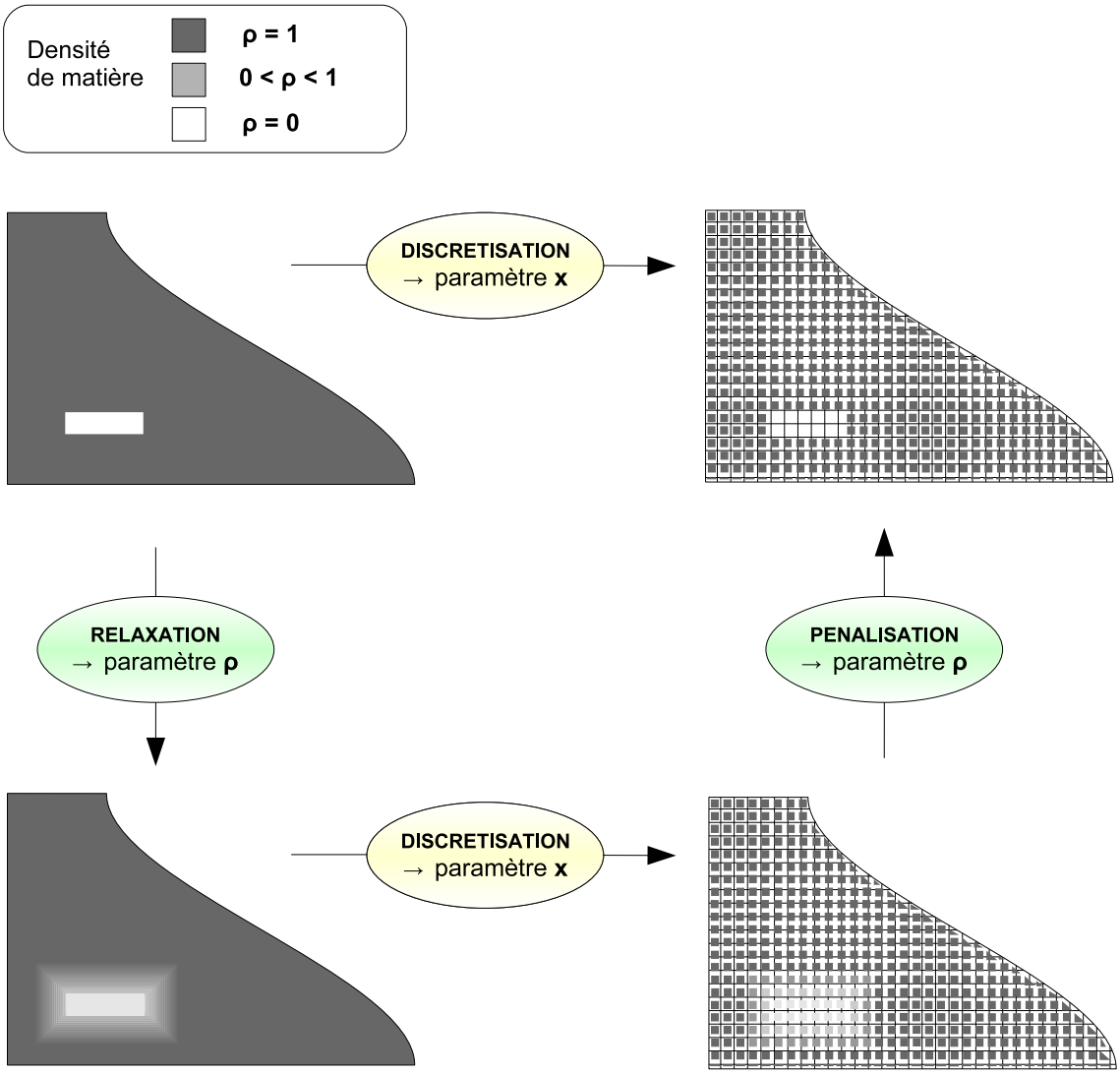


FIG. I.14 – Méthodes de relaxation : illustration

On cherche alors l'influence des variations de la frontière de ω sur la fonction coût J.

Définition Avec les notations de la section 4.1, on considère

- ω_0 un domaine de départ,
- $\omega = (Id + \theta)(\omega_0)$, avec $\theta \in C^1(\mathbb{R}^d, \mathbb{R}^d)$, le domaine déformé.

Le gradient de forme de J en ω_0 est la différentielle (au sens de Fréchet) notée $D_F J(\omega_0)$ de la fonction $\theta \mapsto J((Id + \theta)(\omega_0))$ en 0, c'est-à-dire :

$$J((Id + \theta)(\omega_0)) = J(\omega_0) + D_F J(\omega_0)(\theta) + o(\theta). \tag{I.18}$$

Son expression est connue pour des fonctions particulières (comme la compliance) mais peut être difficile à calculer.

4.3.2 Méthode de lignes de niveaux

Cette méthode est présentée dans [ADGJT05] : on capture une forme $\omega \subset \Omega$ sur un maillage fixe de Ω comme une ligne de niveau

$$\begin{array}{lll} \Psi : \Omega \to \mathbb{R} \quad \text{avec} & \Psi(x) = 0 & \Leftrightarrow x \in \delta\omega \\ & \Psi(x) < 0 & \Leftrightarrow x \in \omega \\ & \Psi(x) > 0 & \Leftrightarrow x \notin \omega \end{array}$$

Au cours des itérations, la forme $\omega(t)$ va évoluer selon un temps t avec une vitesse $V(x,t)$:
pour améliorer ω, c'est-à-dire faire décroître J, on va transporter sa frontière, c'est à dire la ligne de niveau $\Psi = 0$, suivant la direction opposée au gradient de forme, t joue le rôle de pas de descente.

Cette méthode est classée dans l'optimisation topologique car elle permet la disparition ou fusion de trous existants.

On a alors les caractéristiques :

- avantage : pas de remaillage de la forme nécessaire au cours des itérations,
- inconvénient : l'algorithme conduit généralement à un minimum local, donc la solution dépend fortement de la forme de départ.

Pour créer de nouveaux trous, on peut utiliser cette méthode couplée avec celle du gradient topologique (voir section suivante), qui donne l'information nécessaire.

4.3.3 Gradient topologique

Connue au départ sous le nom de "*bubble method*" définie par Schumacher (voir [EKS94]), cette méthode consiste à introduire un trou de petite taille dans le domaine et à l'agrandir avec une méthode d'optimisation de forme classique.
La justification mathématique a été apportée par Sokolowski (voir [SZ99]).

Définition Données :

- $\omega \subset \mathbb{R}^d$, $d = 2, 3$ la forme,
- $\delta \subset \mathbb{R}^d$ un trou contenant l'origine,
- $x_0 \in \omega$, $\rho > 0$
- $\omega_\rho = \omega \setminus \overline{\delta_\rho}$ avec $\delta_\rho = x_0 + \rho\ \delta$, le domaine perforé.

On dit que la fonction $J : \Omega \to \mathbb{R}$ admet un développement asymptotique topologique si

$$J(\omega_\rho) = J(\omega) + \rho^d\ D_T J(x_0) + o(\rho^d). \tag{I.19}$$

alors $D_T J(x_0)$ est appelé le **gradient topologique** de J au point x_0.

Intérêt Le gradient topologique donne une information sur l'opportunité de faire apparître un trou à un endroit de la structure :
soit J à minimiser, si $D_T J(x_0) < 0$ alors créer un trou en x_0 fait décroître J.

Calcul Le professeur Masmoudi, du laboratoire MIP, a développé une méthode adjointe qui permet de calculer l'expression du gradient topologique pour des fonctions coûts générales (voir [Mas01], [GGM01]).

Pour cela, on utilise :

- la méthode du lagrangien généralisé,
- la troncature de domaine.

Cette méthode est en général peu coûteuse, mais le calcul du gradient topologique peut être difficile ; on peut utiliser dans ce cas une approximation.
On connaît son expression dans le cas de l'élasticité linéaire ou en électromagnétisme pour les équations de Maxwell (voir [CGGM00], [Sam04],).

Utilisation Exemple d'algorithme :

- Initialisation : forme $\omega_k = \omega_0$
- Tant que le test d'arrêt est non satisfait
  - calculer u_k et p_k solutions directes et adjointes de l'équation d'état
  - calculer le gradient topologique $D_T^k J(x)$ pour tout $x \in \omega_k$
  - $\omega_{k+1} = \{x \in \Omega_k ; D_T^k J(x) \geq 0\}$
  - $k = k + 1$

En pratique, on discrétise en espace selon le paramètre x, on dispose donc d'un maillage de conception.
Ses éléments sont classés suivant la valeur du gradient topologique obtenue, c'est-à-dire suivant leur sensibilité à la présence de matiére : les éléments de valeur négative ou la plus faible sont supprimés.

Résultat complémentaire Soit J de la forme

$$J : L^d(\omega) \to \mathbb{R} \quad , d = 1, 2.$$

On définit :

- La forme linéaire g associée à $J'(c)$ par $J'(c)\ \delta c = \int_\Omega g\ \delta c$,
  et on suppose que g est régulière.

- Une perturbation topologique dans une boule centrée en x_0 par

$$\delta c_\epsilon = \begin{cases} \delta & \text{dans} B(x_0, \epsilon) \\ 0 & \text{ailleurs.} \end{cases} \tag{I.20}$$

- La mesure de la boule $\rho(\epsilon) = mes(B(x_0, \epsilon))$.

Sous ces conditions, on a

$$J(c + \delta c_\epsilon) = J(c) + g(x_0)\ \delta\ \rho(\epsilon) + o(\rho(\epsilon)). \tag{I.21}$$

On en déduit que dans ce cas le gradient topologique $D_T J$ coincide avec le gradient classique J'.

Chapitre II

Optimisation des lois d'alimentation

Sommaire

1 Introduction

En vertu du principe de découplage des problèmes, (voir section 2.2.3 du chapitre I page 21) nous allons d'abord résoudre le problème de **synthèse de réseau**, ou optimisation des lois d'alimentations.

Nous cherchons la loi d'alimentation d'une antenne réseau, c'est-à-dire un **vecteur complexe qui indique la phase et le module qui seront appliqués à chaque ER**, et qui permet de **vérifier le gabarit** demandé, c'est-à-dire les **contraintes de rayonnement** (voir en section 2.2.1 page 18).

La méthode générale consiste à définir une fonction qui mesure l'**erreur** entre le **rayonnement** de l'antenne pour une alimentation donnée et le gabarit : il faut alors **minimiser** cette fonction coût.

Le rayonnement est une **fonction explicite** de type quadratique : nous pouvons ainsi facilement **calculer** son **gradient** et utiliser des méthodes d'optimisation performantes.

Cependant, ce problème d'optimisation est difficile, avec un **grand nombre de minima locaux** (voir [Dub97] et [DA97]). En effet, la dimension de l'espace de recherche est élevée par rapport aux problèmes de synthèse de réseau classiques, puisque l'antenne de départ comporte plusieurs centaines d'ER.

Nous avons choisi l'approche suivante :

- Il est naturel pour un ingénieur de travailler avec le module et la phase de l'alimentation : cependant, la phase introduit une forte non-linéarité à cause de l'exponentielle complexe.

  Pour **réduire** cette **non-linéarité**, nous prenons alors comme **variables d'optimisation** les **parties réelle et imaginaire** des composantes de l'alimentation.

  De la même façon, la conversion en décibels est fortement non-linéaire à cause du logarithme, nous travaillerons donc avant cette conversion.

- L'**idée fondamentale** suivante est de **reformuler le problème sous forme simple** avant d'appliquer l'algorithme d'optimisation : nous utilisons le **principe de conservation de l'énergie pour modifier les contraintes de rayonnement et rendre le problème convexe**.

  La **solution unique** du problème transformé est alors le **minimum global du problème initial**.

2 Description du problème

2.1 Données

Eléments rayonnants élémentaires : ERel L'antenne de départ (voir figure II.1) est constituée d'un réseau plan régulier de n éléments rayonnants élémentaires, notés **ERel** dans la suite, remplissant un disque.

Eléments rayonnants : ER Les ERel sont considérés comme des briques carrées de taille fixe servant à former des éléments rayonnants rectangulaires, notés **ER** dans la suite, par regroupements.

Nous obtenons alors une antenne à réseau plan irrégulier de n ER (voir figure II.2).

Alimentation a Les ERel ou les ER sont contrôlés chacun avec un module et une phase, et l'alimentation de l'antenne est notée $a = (a_1, \ldots, a_n)^T \in \mathbb{C}^n$, avec $a_j = A_j \, e^{i\varphi_j}$.

Directivité $\mathcal{D}$ La superposition des champs rayonnés par les ER nous donne le rayonnement global de l'antenne, évalué par la directivité notée $\boldsymbol{\mathcal{D}}$ qui est calculable en tout point de la Terre.

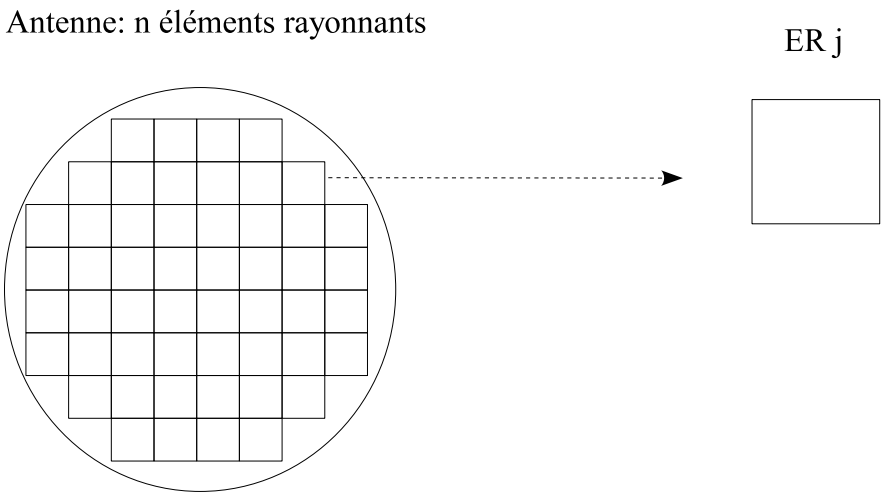


FIG. II.1 – Antenne réseau à maille régulière

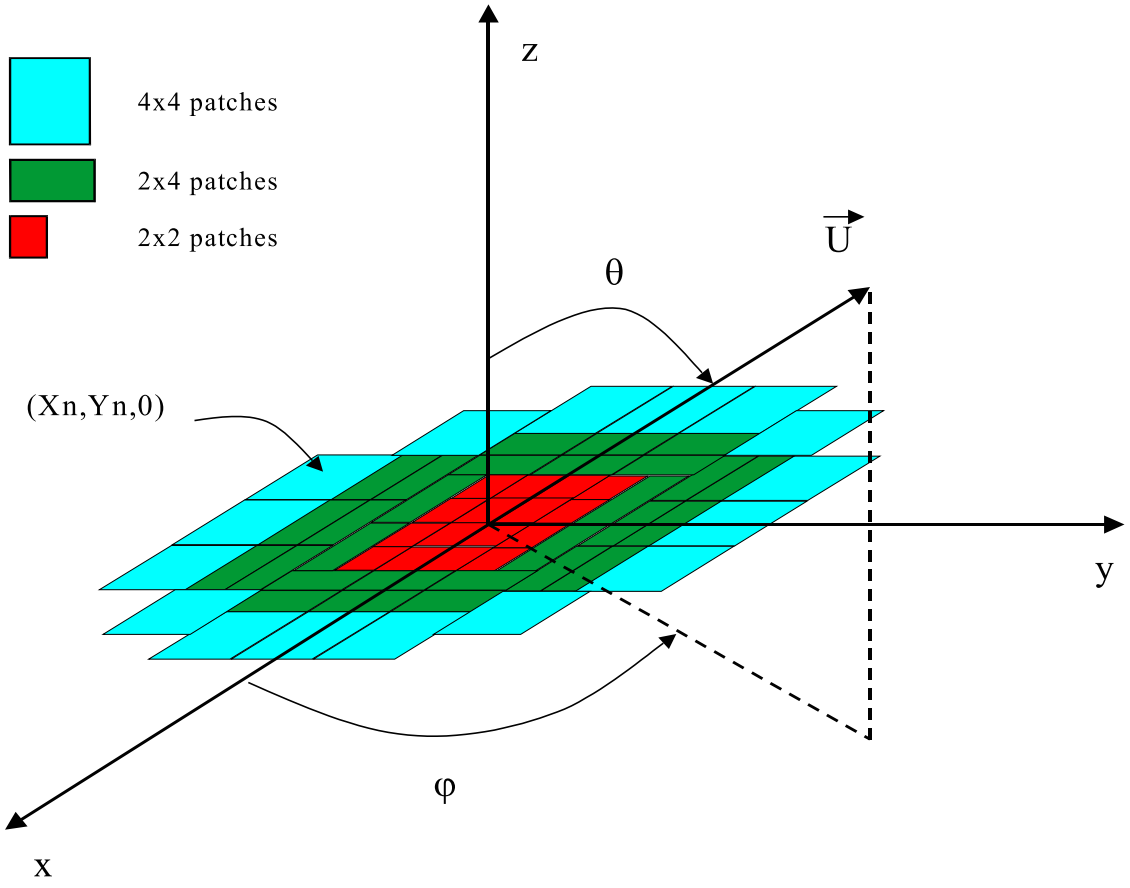


FIG. II.2 – Antenne réseau à maille régulière (trois motifs d'ER différents) dans son repère cartésien

Discrétisation de la Terre On considère qu'on discrétise l'ensemble des directions de rayonnement, et on note $x_j = (x_j, v_j)$, $j \in J_p = \{1 \dots p\}$ les points de

discrétisation (ou de maillage) sur la Terre associés, appelées **stations**.

2.2 Contraintes de rayonnement

La directivité de l'antenne $\mathcal{D}$ doit vérifier un certain nombre de contraintes en chaque point de maillage de la Terre (voir figure II.7) afin de contrôler le rayonnement émis.

Ces contraintes définissent le gabarit de l'antenne, et consistent en un niveau minimum et/ou maximum de directivité dans chaque zone.

2.2.1 Zones de gain et d'isolation

Dans la zone de gain, on cherche à augmenter le rayonnement utile (lobe principal).

Dans la zone d'isolation au contraire, on cherche à diminuer le rayonnement (lobes secondaires).

On note :

- J_{Gain} et J_{Isol} les indices des points de maillage respectivement de la zone de gain et d'isolation,
- G_{Gain} et G_{Isol} les valeurs de rayonnement désirées respectivement sur la zone de gain et d'isolation.

Remarque On peut aussi introduire une zone d'interférence, qui est traitée comme une deuxième zone d'isolation avec une valeur de rayonnement désiré différente.

2.2.2 Modélisation des contraintes

On note $\mathcal{D}(a, x_j)$, ou $\boldsymbol{\mathcal{D}_j(a)}$, la fonction de rayonnement associée à l'alimentation a et au point x_j du maillage Terre.

Sur chaque point de maillage x_j, le niveau de rayonnement désiré est donné par deux valeurs notées

- G_j^m pour le niveau de rayonnement minimum souhaité,
- G_j^M pour le niveau de rayonnement maximum souhaité.

On note $c_j(a)$ la contrainte de gabarit, différence entre le rayonnement calculé $\mathcal{D}_j$ et celui souhaité G_j (on écrit cette différence pour qu'elle soit négative quand la contrainte est respectée).

Gabarit de type intervalle Le gabarit est ici l'intervalle des valeurs admissibles pour le rayonnement, définies en chaque point de maillage Terre.

Mathématiquement, on veut résoudre le problème :

calculer la loi d'alimentation a telle que

$$\forall j \in J_p \quad G_j^m \leqslant \mathcal{D}_j(a) \leqslant G_j^M \iff \forall j \in J_p \quad \begin{cases} c_j(a) = G_j^m - \mathcal{D}_j(a) & \leqslant 0 \\ c_j(a) = \mathcal{D}_j(a) - G_j^M & \leqslant 0. \end{cases} \tag{II.1}$$

C'est à dire le rayonnement $\mathcal{D}$ vérifie le gabarit en tout point du maillage Terre.

On peut représenter ce gabarit à valeurs dans des intervalles par exemple par la figure II.3.

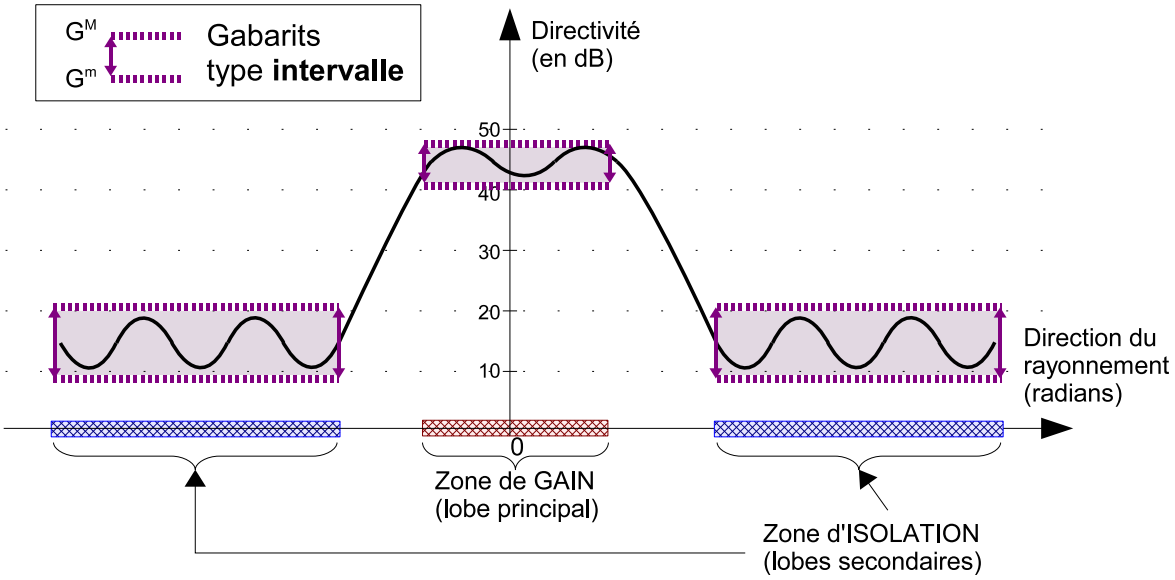


FIG. II.3 – Gabarit à valeurs dans des intervalles

Gabarit de type minimum-maximum Dans ces cas là, le gabarit peut se présenter avec une seule contrainte de type inégalité par point de maillage Terre, et le problème s'écrit :

calculer la loi d'alimentation a tel que

$$\begin{cases} \forall j \in J_{Gain} & c_j(a) = G_{Gain} - \mathcal{D}_j(a) & \leqslant 0 \\ \forall j \in J_{Isol} & c_j(a) = \mathcal{D}_j(a) - G_{Isol} & \leqslant 0 \end{cases} \tag{II.2}$$

avec

- $\forall j \in J_p \; G_{Gain} := G_j^m$,
- $\forall j \in J_p \; G_{Isol} := G_j^M$.

On peut représenter un gabarit à valeurs minimum ou maximum par exemple par la figure II.4.

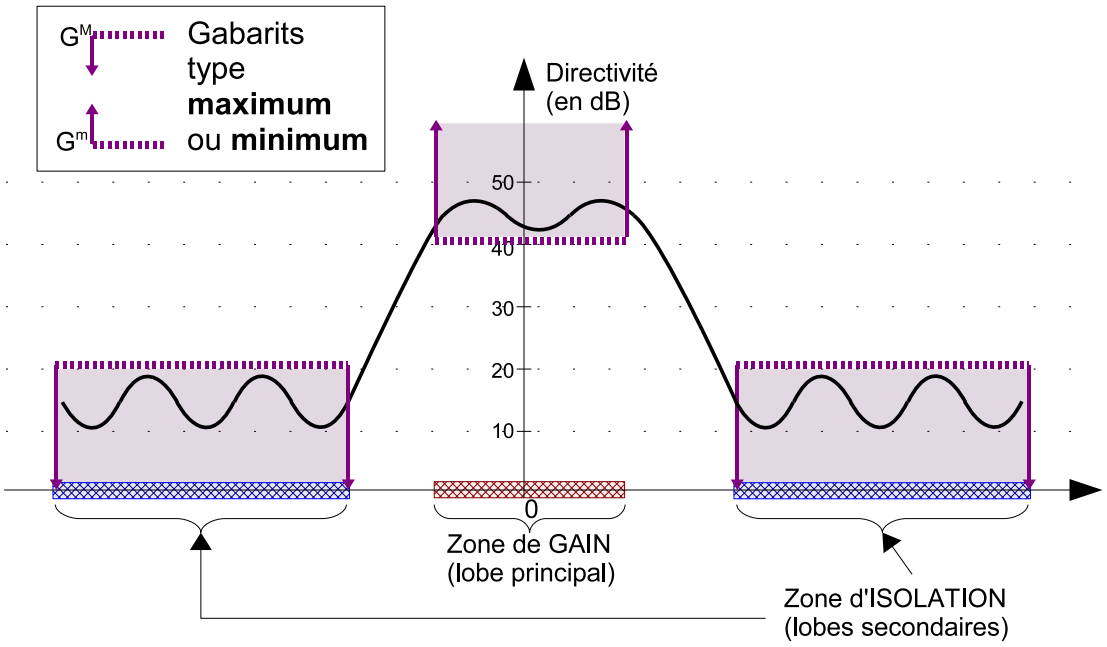


FIG. II.4 – Gabarit à valeurs minimum ou maximum

2.3 Problème d'optimisation

2.3.1 Erreur locale

En un point de maillage x_j fixé, on définit l'erreur locale σ_j par

$$\begin{aligned} \sigma_j := c_j(a)^+ &:= c_j(a) \quad \text{si } c_j(a) \geqslant 0 \\ &:= 0 \qquad \text{si } c_j(a) < 0. \end{aligned} \tag{II.3}$$

Ainsi, l'erreur locale

- reste strictement positive si la contrainte de gabarit n'est pas vérifiée en x_j,
- est nulle si la contrainte de gabarit est vérifiée en x_j.

2.3.2 Erreur globale : fonction coût

On peut alors définir l'erreur globale σ à partir de l'ensemble des erreurs locales

$$\sigma(a) := \|(\sigma_1, \ldots, \sigma_p)\| \quad , \sigma_j \in J_p \tag{II.4}$$

avec $\|.\|$ une norme usuelle de $\mathbb{R}^n$. On choisira la formulation adaptée selon le gabarit proposé et le contexte du problème à résoudre.

Ainsi, l'erreur globale

- est strictement positive si la contrainte de gabarit n'est pas vérifiée en tout x_j,
- est nulle si la contrainte de gabarit est vérifiée partout.

Remarque On peut définir de manière analogue :

- l'erreur sur zone de gain $\sigma_{Gain}(a) := \|(\sigma_1, \ldots, \sigma_{p_{Gain}})\| \quad , \sigma_j \in J_{Gain}$,
- l'erreur sur zone d'isolation $\sigma_{Isol}(a) := \|(\sigma_1, \ldots, \sigma_{p_{Isol}})\| \quad , \sigma_j \in J_{Isol}$.

2.3.3 Formulation

On choisit $\sigma(a)$ comme fonction coût pour obtenir un problème d'optimisation : si on minimise sa valeur jusqu'à zéro, on résoud le problème de synthèse de réseau.

$$\begin{cases} \min\limits_{a \in \mathbb{C}^n} \sigma(a) \\ \text{sous des contraintes éventuelles sur } a. \end{cases} \tag{II.5}$$

Parmi les formulations possibles selon la norme choisie pour l'erreur globale, on peut citer les suivantes.

Problème de type minmax

$$\begin{cases} \min\limits_{a \in \mathbb{C}^n} \left(\max\limits_{j \in J_p} \sigma_j \right) \\ \text{sous des contraintes éventuelles sur } a. \end{cases} \tag{II.6}$$

On contrôle au mieux le rayonnement car on maîtrise le maximum de l'écart entre les diagrammes calculé et souhaité. Mais la fonction coût n'est pas différentiable.

Problème de type moindres carrés

$$\begin{cases} \min\limits_{a \in \mathbb{C}^n} \left(\sum\limits_{j \in J_p} \sigma_j^2 \right) \\ \text{sous des contraintes éventuelles sur } a. \end{cases} \tag{II.7}$$

Dans certains cas, le rayonnement peut osciller car on ne maîtrise pas forcément les écarts importants qui peuvent apparaître dans certaines directions. Mais ici la fonction coût est différentiable : nous pourrons utiliser des méthodes d'optimisation efficaces.

Pondération On peut chaque fois que l'on considère plusieurs critères d'erreur à minimiser, au niveau local avec σ_j ou au niveau des zones avec σ_{Gain} et σ_{Isol}, leur associer des poids ou coefficients de pondération, qui sont des constantes positives.

On peut ainsi augmenter la contribution de l'un des critères d'erreur à la fonction coût en choisissant un poids plus important que les autres, et ainsi privilégier la satisfaction de la contrainte associée.

Exemple $\sigma(a) = \alpha_1 \, \sigma_{Gain} + \alpha_2 \, \sigma_{Isol}$ avec $(\alpha_1, \alpha_2) \in \mathbb{R}^{+*} \times \mathbb{R}^{+*}$.

En choisissant α_1 grand par rapport à α_2, on cherchera à satisfaire en priorité la contrainte sur la zone de gain.

2.3.4 Résolution

Dans le cas général d'une synthèse de réseau, quand on a tranformé les contraintes de gabarit en une fonction coût, on doit résoudre un problème d'optimisation globale éventuellement sous contraintes.

En fonction des caractéristiques du problème à résoudre, on choisit alors une des méthodes présentées dans la section 3 du chapitre I.

C'est ce que nous allons faire pour les deux études présentées dans la suite.

3 Etude préliminaire

3.1 Introduction

Thalès a d'abord fourni au laboratoire MIP une étude de faisabilité sur un cas test que nous allons présenter dans cette section.

L'objectif était de vérifier si une méthode algorithmique pouvait réduire significativement le nombre de contrôles utilisés dans l'antenne en regroupant des éléments rayonnants, tout en continuant à satisfaire au mieux les contraintes.

Ce cas test nous a permis d'établir la méthode de calcul des lois d'alimentation optimales.

Ici, la formule explicite de la directivité $\mathcal{D}$ comme quotient permet d'éviter une contrainte sur l'alimentation a : pour obtenir une alimentation physiquement réalisable, il suffit de normer la solution obtenue.

Seules les contraintes de rayonnement sont à respecter.

L'idée majeure est de transformer, grâce au principe de conservation de l'énergie, un problème difficile de maximisation de la directivité en un problème convexe de minimisation : au lieu maximiser le rayonnement à l'intérieur de la zone utile, qui est un spot fin, nous allons le minimiser à l'extérieur.

3.2 Données

Nous présentons maintenant les hypothèses du problème physique à traiter.

3.2.1 Antenne

L'antenne est définie par un réseau de n ER rectangulaires qui remplissent un disque de diamètre D_{ant}.

Pour chaque ER d'indice j (voir figure II.5) :

- son centre par rapport au centre O de l'antenne est noté M_j,
- il est constitué de sources élémentaires appelées patchs :
  - Q_j patches en hauteur,
  - P_j patches en longueur,
  - Q_j et P_j des puissances de deux,
  - la distance entre deux patchs est $d = 0.9 \cdot \lambda$, avec λ la longueur d'onde.

L'antenne de départ est une antenne formée d'un réseau à maille régulière d'ERel carrés.

3.2.2 Directivité

Alimentation On associe à chaque ER $j \in 1, \ldots, n$ un module A_j et une phase φ_j.

La loi d'alimentation de l'antenne est notée $a = (a_1, \ldots, a_n)^T \in \mathbb{C}^n$, avec $a_j = A_j\, e^{i\,\varphi_j}$.

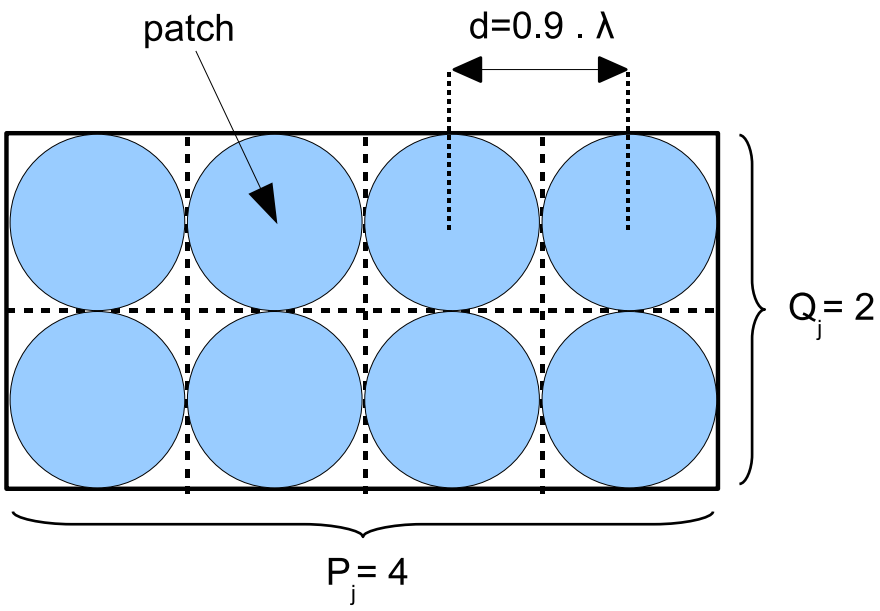


FIG. II.5 – Exemple d'ER 4x2 patches

Champ d 'un ER On note $x \in \mathbb{R}^2$ un point de la Terre :

- $x = (u, v)^T$ les coordonnées cartésiennes utilisées pour les calculs,
- $x = (r, \theta, \varphi)$ les coordonnées sphériques utilisées pour les définitions,
- (θ, φ) est la direction correspondante à $\overrightarrow{Ox}$ (voir figure II.2).

Le champ rayonné par un patch est donné par

$$e(x) = \cos\left(\frac{\pi\theta}{2\theta_1}\right)^{\alpha}. \tag{II.8}$$

Le champ rayonné par l'ER j est donné par

$$E_j(x) = \frac{\sin(\psi_u P_j)}{\sin(\psi_u)} \frac{\sin(\psi_v Q_j)}{\sin(\psi_v)}\, e(x) \tag{II.9}$$

avec

$$\psi_u = \frac{\pi d u}{\lambda} \quad \text{et} \quad \psi_v = \frac{\pi d v}{\lambda}. \tag{II.10}$$

Ces formules sont déduites des équations de Maxwell qui régissent les phénomènes de propagation des ondes électromagnétiques (voir [Bal97]).

Intéractions entre sources Nous utilisons un hypothèse de travail fondamentale : les intéractions entre sources sont négligées.

Champ du réseau Dans ce cadre, on peut appliquer un théorème de superposition des champs (voir [Tho90]) à l'ensemble des sources élémentaires, qui sont les patches, pour calculer le champ E, ou diagramme de rayonnement, du réseau en fonction de $e(x)$:

$$E(a,x) = \sum_{j=1}^{n} \sum_{k=1}^{n_j} a_j \; e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{ON_k}, \overrightarrow{Ox}\rangle} e(x). \tag{II.11}$$

avec

- $n_j = P_j \times Q_j$ le nombre de patches constituant l'ER j,
- N_k le centre du patch k dans l'ER j.

Après les calculs présentés en annexe 2, la formule se simplifie et on obtient l'expression suivante :

$$E(a,x) = \sum_{j=1}^{n} a_j \; e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{OM_j}, \overrightarrow{Ox}\rangle} \; E_j(x). \tag{II.12}$$

On peut donc exprimer le champ total comme une combinaison linéaire à coefficients complexes des champs des ER.

On obtient la fonction champ

$$\begin{array}{rccl} E: & \mathbb{C}^n \times \mathbb{R}^2 & \longrightarrow & \mathbb{C} \\ & (a,x) & \longmapsto & E(a,x) = \displaystyle\sum_{j=1}^{n} a_j \; \widehat{E_j}(x) \end{array} \tag{II.13}$$

avec $\widehat{E_j}(x) := e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{OM_j}, \overrightarrow{Ox}\rangle} E_j(x) \; \in \mathbb{C}$;

Directivité Nous rappelons qu'on calcule la directivité en prenant l'énergie (ou puissance) rayonnée dans une direction divisée par l'énergie totale.

La directivité de l'antenne est donnée par la fonction

$$\begin{array}{rccl} \mathcal{D}: & \mathbb{C}^n \times \mathbb{R}^2 & \longrightarrow & \mathbb{R} \\ & (a,x) & \longmapsto & \mathcal{D}(a,x) = 4\pi \; \dfrac{|E(a,x)|^2}{\iint_\Omega |E(a,x)|^2 \sin\theta \; d\theta d\varphi}. \end{array} \tag{II.14}$$

Exprimée en décibels, nous la noterons $\widetilde{\mathcal{D}} := 10.\log_{10}(\mathcal{D}) = 10 \; \frac{\ln(\mathcal{D})}{\ln(10)}$.

La fonction directivité intervient dans la fonction coût, et nous pouvons calculer explpicitement son gradient (voir calcul en annexe 2).

Normalisation en puissance La directivité $\mathcal{D}$ vérifie la propriété suivante : comme le champ E est linéaire en a, $\mathcal{D}$ est constante par homothétie

$$\forall \lambda \in \mathbb{R} \quad \mathcal{D}(\lambda\ a) = \mathcal{D}(a). \tag{II.15}$$

Or l'alimentation recherchée est physiquement réalisable pour une puissance donnée, par exemple $\|a\| = 1$.

Chaque composante de a correspond à un dispositif électronique de contrôle branché derrière chaque ER, et la puissance est ensuite répartie sur chaque contrôle, et définit ainsi les modules de a.

Nous pouvons donc ici travailler sans contrainte sur a, il suffit de normaliser l'alimentation à la fin du processus d'optimisation.

Propriété quadratique convexe L'énergie totale rayonnée par l'antenne est constante pour une alimentation a de norme (ou puissance) fixée.

Dans l'expression précédente, le dénominateur est constant, et nous obtenons

$$\mathcal{D}(a,x) = K\ |E(a,x)|^2 = K\ \left|\sum_{j=1}^{n}\ a_j\ \widehat{E_j}(x)\right|^2 \tag{II.16}$$

avec K constante réelle.

La directivité $\mathcal{D}(a)$ s'écrit donc comme un module au carré, et si nous exprimons $a_j = (x_{a_j}, y_{a_j})$ sous forme de partie réelle et imaginaire, nous obtenons une **fonction de type quadratique et convexe** par rapport aux composantes de a.

3.2.3 Zones sur la Terre

On définit les zones suivantes pour les contraintes de rayonnement (voir tableau II.1 et figure II.6) :

avec $(R_t, R_u, R_v, R_c, R_{sp}) \in (\mathbb{R}^{+*})^5$ des constantes données.

La zone de couverture est une ellipse, et son centre (U_1, V_1) est la direction dans laquelle pointe l'antenne.

Ces zones seront discrétisées par un maillage qui n'est pas imposé à priori. Il sera composé des points notés $x_k = (u_k, v_k)^T$, $k \in J_p = \{1, \ldots, p\}$.

3.2.4 Contraintes de rayonnement

Le gabarit est défini par les contraintes suivantes (tableau II.2) pour un spot k fixé :

Les **zones de gabarit** sont **disjointes**, ainsi à un point quelconque sur la Terre on associe une unique contrainte de gabarit.

Les contraintes de gabarit s'écrivent alors :

Zone	Notation	Définition pour $x = (u, v)^T$
Spot k	Φ_0	$(u - u_k)^2 + (v - v_k)^2 \leqslant R_{sp}^2$
Couronne autour du spot k	Φ_1	$R_{sp}^2 < (u - u_k)^2 + (v - v_k)^2 \leqslant R_c^2$
Zone de couverture	Φ_2	$\left(\frac{u-U_1}{R_u}\right)^2 + \left(\frac{v-V_1}{R_v}\right)^2 \leqslant 1$
Terre	Φ_3	$(u^2 + v^2) \leqslant R_t^2$
Espace total	Φ_4	

TAB. II.1 – Zones sur la Terre pour les contraintes

Zone de gabarit	Notation	Objectif
Zone utile	Φ_0	Puissance maximale
Zone d'interférence	$\Phi_2 \setminus (\Phi_0 \cup \Phi_1)$	Isolation entre spots
Zone d'isolation	$\Phi_3 \setminus \Phi_2$	Eviter les remontées de lobes de réseaux
Hors Terre	$\Phi_4 \setminus \Phi_3$	Eviter les pertes par débordement

TAB. II.2 – Contraintes de rayonnement

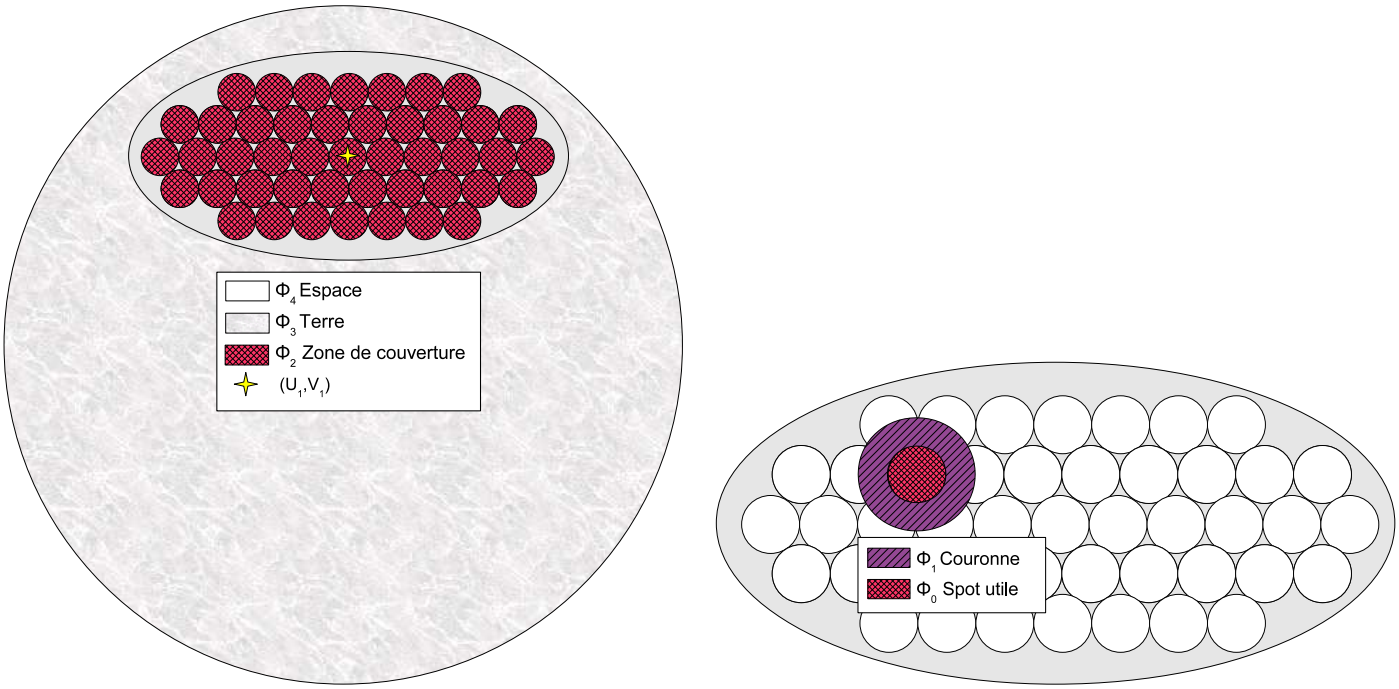


FIG. II.6 – Domaines sur la Terre

$$\begin{cases} \forall x \in \Phi_0 & \widetilde{g_0} \leqslant \widetilde{\mathcal{D}}(a,x) \leqslant \widetilde{g_0} + \widetilde{d_0} \\ \forall x \in \Phi_2 \setminus (\Phi_0 \cup \Phi_1) & \widetilde{\mathcal{D}}(a,x) \leqslant \widetilde{g_0} - \widetilde{d_1} \\ \forall x \in \Phi_3 \setminus \Phi_2 & \widetilde{\mathcal{D}}(a,x) \leqslant \widetilde{g_0} - \widetilde{d_2} \\ & \iint_{\Phi_4 \setminus \Phi_3} \widetilde{\mathcal{D}}(a,x)\ dx \leqslant \frac{10}{100} \iint_{\Phi_4} \widetilde{\mathcal{D}}(a,x)\ dx. \end{cases} \tag{II.17}$$

Les valeurs $\widetilde{g_0}, \widetilde{d_0}, \widetilde{d_1}, \widetilde{d_2} \in (\mathbb{R}^{+*})^4$ positives et données en décibels définissent le gabarit, qui doit correspondre aux capacités de l'antenne pour pouvoir être réalisé.

La première contrainte signifie qu'on veut une puissance de rayonnement maximale dans la zone de gain, qui est le spot Φ_0.

Les deux contraintes suivantes signifient qu'on veut abaisser la puissance respectivement

- dans la zone d'interférence, qui est la zone de couverture, hors spot Φ_0 entouré de la couronne Φ_1,
- dans la zone d'isolation, qui est la Terre hors zone de couverture.

La dernière contrainte ne dépend pas de x et signifie qu'on veut la puissance hors Terre inférieure à 10% de la puissance totale.

3.3 Modélisation et idées clés

3.3.1 Idée clé : réduire la non linéarité

Conversion en décibels La conversion de la directivité en décibels afin d'appliquer les contraintes de gabarit définissant le problème, permet d'obtenir des valeurs plus facilement interprétables, qui varient sur une plus petite échelle.

Mais du point de vue des propriétés mathématiques de la fonction directivité $\widetilde{\mathcal{D}}$ qui définit la fonction coût, on applique à l'expression de $\mathcal{D}$ (voir section 3.2.2) un logarithme, qui est une application non linéaire, et qui va donc faire perdre la propriété quadratique en a : or une fonction de type quadratique a de bonnes propriétés en vue d'une minimisation.

L'idée est donc de ne convertir en décibels qu'à la fin de l'algorithme d'optimisation, uniquement pour comparer avec les valeurs d'origine du gabarit données en décibels.

On note

- $\widetilde{\mathcal{D}}$ la directivité exprimée en dB, utilisée pour vérification et affichage des résultats,
- $\mathcal{D}$ la directivité avant la conversion en dB, utilisée pour les calculs.

  On applique $\widetilde{\mathcal{D}}(a) = e^{\mathcal{D}(a)\frac{\ln 10}{10}}$ pour retrouver $\widetilde{\mathcal{D}}$ à partir de $\mathcal{D}$.

De même pour les valeurs de gabarit utilisées dans les calculs, on change les valeurs données en leur apliquant la transformation inverse de la conversion en décibels.

On note :

- $g_0 = e^{\widetilde{g_0}\frac{\ln 10}{10}}$
- $d_j = e^{\widetilde{d_j}\frac{\ln 10}{10}}$, $j = 0, 1, 2$

Alimentation a en partie réelle, partie imaginaire D'un point de vue physique, il est naturel d'exprimer le vecteur alimentation en fonction du module et de la phase de chaque ER :

- $a = (A_1\ e^{i\,\varphi_1}, \ldots, A_n\ e^{i\,\varphi_n}) \in \mathbb{C}^n$,
- ou $a = (A_1, \ldots, A_n \vdots \varphi_1 \ldots, \varphi_n) \in \mathbb{R}^{2n}$,

Nous utiliserons ces formes pour vérification et affichage sous forme de cartes des résultats.

Mais l'exponentielle complexe, de la forme $e^{i\ \varphi_k} = cos(\varphi_k)\ + i\ sin(\varphi_k)$, introduit avec les fonctions trigonométriques une partie fortement non linéaire par rapport à la phase φ_k dans l'expression de la directivité.

D'un point de vue mathématique, pour conserver la propriété quadratique de $\mathcal{D}$ comme précédemment, il vaut mieux travailler pour les calculs d'optimisation avec le vecteur alimentation en partie réelle et partie imaginaire, sous la forme $a = (x_{a_1}, \ldots, x_{a_n} \vdots y_{a_1}, \ldots, y_{a_n}) \in \mathbb{R}^{2n}$ avec $a_j = x_{a_j} + i\ y_{a_j}$.

Dans la partie optimisation topologique, nous utiliserons les modules des composantes de a : nous effectuerons cette conversion non linéaire du type $|a_j| = x_{a_j}^2 + y_{a_j}^2$ après l'algorithme d'optimisation.

Conclusion On peut résumer l'idée clé de réduire la non linéarité du problème par le schéma suivant :

$$
\begin{array}{ccccc}
\textit{module et phase} & & & & \textit{en dB} \\
a = (\dots Aj\ e^{i\varphi_j} \dots) & \longrightarrow & \textsc{Optimisation classique} & \longrightarrow & \widetilde{\mathcal{D}}(a) \\
\downarrow & & & & \uparrow \\
\text{NON LINÉAIRE} & & & & \text{NON LINÉAIRE} \\
\downarrow & & & & \uparrow \\
\tilde{a} = (\dots x_{a_j} + i\ y_{a_j} \dots) & \longrightarrow \text{QUADRATIQUE} \longrightarrow & & |E(a)|^2 \ \rightarrow & \mathcal{D}(a) \\
\underbrace{\textit{partie réelle et imaginaire}} & & & & \underbrace{\textit{avant conversion}} \\
\multicolumn{5}{c}{\textbf{OPTIMISATION avec REDUCTION de la NON LINEARITE}}
\end{array}
$$

3.3.2 Définition des contraintes

Comme dans la section 2.2.2, on modélise les contraintes de gabarit sous forme d'inégalités.

Première formulation Pour une antenne à géométrie donnée (n ER) et pour un spot k fixé, le problème à résoudre est :

trouver $a \in \mathbb{C}^n$ tel que

$$
\left(\widehat{\mathcal{P}_0}\right) \quad \left\{ \begin{array}{lll}
\forall\ x \in \Phi_0 & c_1(a,x) = -\mathcal{D}(a,x) + g_0 & \leqslant 0 \\
\forall\ x \in \Phi_0 & c_2(a,x) = \mathcal{D}(a,x) - g_0 - d_0 & \leqslant 0 \\
\forall\ x \in \Phi_2 \setminus (\Phi_0 \cup \Phi_1) & c_3(a,x) = \mathcal{D}(a,x) - g_0 + d_1 & \leqslant 0 \\
\forall\ x \in \Phi_3 \setminus \Phi_2 & c_4(a,x) = \mathcal{D}(a,x) - g_0 + d_2 & \leqslant 0 \\
 & \widehat{c_5}(a) = \iint_{\Phi_4 \setminus \Phi_3} \mathcal{D}(a,x)\ dx - \frac{10}{100} \iint_{\Phi_4} \mathcal{D}(a,x)\ dx & \leqslant 0.
\end{array} \right. \tag{II.18}
$$

Modification de la contrainte globale La contrainte $\widehat{c_5}$ est la seule qui soit de nature globale, nous la transformons en contrainte locale c_5 appliquée aux points hors Terre pour l'exprimer de la même manière que c_3 et c_4.

Nous prenons par exemple pour limiter le rayonnement à l'extérieur de la Terre

$$
\forall\ x \in \Phi_4 \setminus \Phi_3 \quad c_5(a,x) = \mathcal{D}(a,x) - \frac{g_0}{10} \leqslant 0. \tag{II.19}
$$

Il restera à vérifier à la fin de l'algorithme que la contrainte globale c_5 est respectée.

Deuxième formulation Pour une antenne à géométrie donnée (n ER) et pour un spot k fixé, le problème s'écrit avec la modification de la contrainte globale :

trouver $a \in \mathbb{C}^n$ tel que

$$
(\mathcal{P}_0) \quad \left\{ \begin{array}{lll}
\forall\ x \in \Phi_0 & c_1(a,x) = -\mathcal{D}(a,x) + g_0 & \leqslant 0 \\
\forall\ x \in \Phi_0 & c_2(a,x) = \mathcal{D}(a,x) - g_0 - d_0 & \leqslant 0 \\
\forall\ x \in \Phi_2 \setminus (\Phi_0 \cup \Phi_1) & c_3(a,x) = \mathcal{D}(a,x) - g_0 + d_1 & \leqslant 0 \\
\forall\ x \in \Phi_3 \setminus \Phi_2 & c_4(a,x) = \mathcal{D}(a,x) - g_0 + d_2 & \leqslant 0 \\
\forall\ x \in \Phi_4 \setminus \Phi_3 & c_5(a,x) = \mathcal{D}(a,x) - \frac{g_0}{10} & \leqslant 0.
\end{array} \right. \tag{II.20}
$$

3.3.3 Reformulation : maximisation sous contraintes

La contrainte c_2 n'est pas essentielle, car si la solution dépasse la valeur $g_0 + d_0$ dans le spot en vérifiant les autres contraintes, elle est acceptable.

Le problème revient donc à maximiser le rayonnement à l'intérieur du spot Φ_0 (contrainte c_1) sous les contraintes c_3 à c_5.

Pour un point x donné, on note $c_{i_x}(a, x)$, $i_x \in \{1, \ldots, 5\}$ la contrainte associée parmi les cinq précédentes, selon la zone à laquelle appartient x.

On peut maintenant énoncer le problème sous la forme d'un problème de maximisation sous contraintes :

$$(\mathcal{P}_1) \quad \begin{cases} \max\limits_{a \in \mathbb{C}^n} \min\limits_{x \in \Phi_0} \mathcal{D}(a, x) \\ \\ \forall\, x \in \Phi \setminus \Phi_0 \quad c_{i_x}(a, x) \leqslant 0. \end{cases} \tag{II.21}$$

3.3.4 Idée clé : conservation de l'énergie

Le problème précédent est difficile, avec un nombre de minima locaux estimé à 2^n car la maximisation d'une fonction de type quadratique convexe n'a pas de bonnes propriétés.

L'idée est de tranformer ainsi le problème précédent en un problème plus simple.

Or d'après le principe de conservation de l'énergie, si on arrive à baisser la valeur du rayonnement partout hors du spot, il va alors se concentrer à l'intérieur de ce dernier.

Maximiser le minimum du rayonnement $\mathcal{D}$ à l'intérieur du spot Φ_0 revient donc à **minimiser le maximum du rayonnement $\mathcal{D}$ à l'extérieur du spot Φ_0**, et ce dernier problème est **convexe** (voir annexe 1), car $\mathcal{D}$ est quadratique convexe.

3.3.5 Idée clé : reformulation sous forme de minimisation convexe

D'après l'idée précédente, le problème de maximisation $(\mathcal{P}_1)$ peut alors aussi s'énoncer sous la forme du problème de minimisation sous contraintes $(\mathcal{P}_2)$:

$$(\mathcal{P}_2) \quad \begin{cases} \min\limits_{a \in \mathbb{C}^n} \max\limits_{x \in \Phi \setminus \Phi_0} \mathcal{D}(a, x) \\ \\ \forall\, x \in \Phi \setminus \Phi_0 \quad c_{i_x}(a, x) \leqslant 0. \end{cases} \tag{II.22}$$

La **minimisation d'une fonction de type quadratique convexe** est un **problème convexe** qui admet une seul minimum : **l'unique solution** de $(\mathcal{P}_2)$ est le **minimum global** recherché en $(\mathcal{P}_1)$.

3.3.6 Reformulation des contraintes

Nous allons associer des contraintes locales au problème de minimisation $(\mathcal{P}_2)$, pour revenir à une forme similaire à la première formulation $(\mathcal{P}_0)$ de la section 3.3.2, pour laquelle nous savons construire une fonction coût à minimiser.

Avec $(\mathcal{P}_2)$, la contrainte de maximisation dans le spot c_1 de $(\mathcal{P}_0)$ disparait, et la minimisation du maximum du rayonnement correspond aux contraintes de minimisation hors du spot c_3 à c_5.

Le problème devient :

trouver $a \in \mathbb{C}^n$ tel que

$$(\mathcal{Q}_0) \quad \begin{cases} \forall\, x \in \Phi_2 \setminus (\Phi_0 \cup \Phi_1) & c_3(a,x) & = \mathcal{D}(a,x) - g_0 + d_1 & \leqslant 0 \\ \forall\, x \in \Phi_3 \setminus \Phi_2 & c_4(a,x) & = \mathcal{D}(a,x) - g_0 + d_2 & \leqslant 0 \\ \forall\, x \in \Phi_4 \setminus \Phi_3 & c_5(a,x) & = \mathcal{D}(a,x) - \frac{g_0}{10} & \leqslant 0. \end{cases} \tag{II.23}$$

Le problème $(\mathcal{Q}_0)$ et le problème de départ $(\mathcal{P}_0)$ ne sont pas strictement équivalents d'un point de vue mathématique, mais leurs solutions sont numériquement proches.

$(\mathcal{Q}_0)$ présente l'avantage d'être plus simple à résoudre que $(\mathcal{P}_0)$, car la fonction coût associée présente de meilleures propriétés pour la minimisation.

3.3.7 Mise en oeuvre

Contrôle du rayonnement autour du spot Avec les contraintes précédentes, le rayonnement n'est pas contrôlé dans le spot et la couronne autour de celui-ci. Or lors des premiers test numériques, cette zone à l'extérieur du spot est apparue trop large pour concentrer correctement le rayonnement à l'intérieur du spot.

Nous rajoutons donc une contrainte de type isolation, ce qui ne change pas la nature du problème $(\mathcal{Q}_0)$ à résoudre. Nous imposons une valeur inférieure à g_0 c'est-à-dire $g_0 - e_0$ comme valeur maximum pour les points autour du spot Φ_0 :

$$\forall\, x \in \widetilde{\Phi_1} \quad \mathcal{D}(a,x) \leqslant g_0 - e_0 \tag{II.24}$$

avec

- $\widetilde{\Phi_1}$ la couronne autour du spot utilisée pour contrôler le rayonnement (moins large que Φ_1 la couronne du gabarit, voir figure II.7),
- $e_0 \in \mathbb{R}^+$ une valeur équivalente à deux ou trois dB.

Après minimisation, on veut cette contrainte saturée, c'est-à-dire

$$\forall\, x \in \widetilde{\Phi_1} \quad \mathcal{D}(a,x) = g_0 - e_0. \tag{II.25}$$

Comme on aura "écrasé" le rayonnement partout à l'extérieur, il va naturellement se concentrer dans la zone où on n'avait pas imposé de contraintes, c'est-à-dire à l'intérieur du spot Φ_0, et sa valeur sera supérieure à celle atteinte sur la couronne $\widetilde{\Phi_1}$ qui définit le bord de Φ_0.
On obtient alors la valeur minimale demandée g_0 dans Φ_0.

Contraintes finales En ajoutant la contrainte précédente sur la couronne de contrôle, et en renommant les contraintes C_j, le problème s'écrit finalement :

trouver $a \in \mathbb{C}^n$ tel que

$$(\mathcal{Q}_1) \quad \begin{cases} \forall\, x \in \Psi_1 & C_1(a,x) & = \mathcal{D}(a,x) - g_0 + e_0 & \leqslant 0 \\ \forall\, x \in \Psi_2 & C_2(a,x) & = \mathcal{D}(a,x) - g_0 + d_1 & \leqslant 0 \\ \forall\, x \in \Psi_3 & C_3(a,x) & = \mathcal{D}(a,x) - g_0 + d_2 & \leqslant 0 \\ \forall\, x \in \Psi_4 & C_4(a,x) & = \mathcal{D}(a,x) - \frac{g_0}{10} & \leqslant 0 \end{cases} \tag{II.26}$$

avec les notations pour les nouvelles zones où s'appliquent les contraintes (voir figure II.7)

- $\Psi_1 := \widetilde{\Phi_1}$ la couronne de contrôle autour du spot,
- $\Psi_2 := \Phi_2 \setminus (\Phi_0 \cup \Phi_1)$ la zone d'interférence,
- $\Psi_3 := \Phi_3 \setminus \Phi_2$ la zone d'isolation,
- $\Psi_4 := \Phi_4 \setminus \Phi_3$ la zone hors Terre.

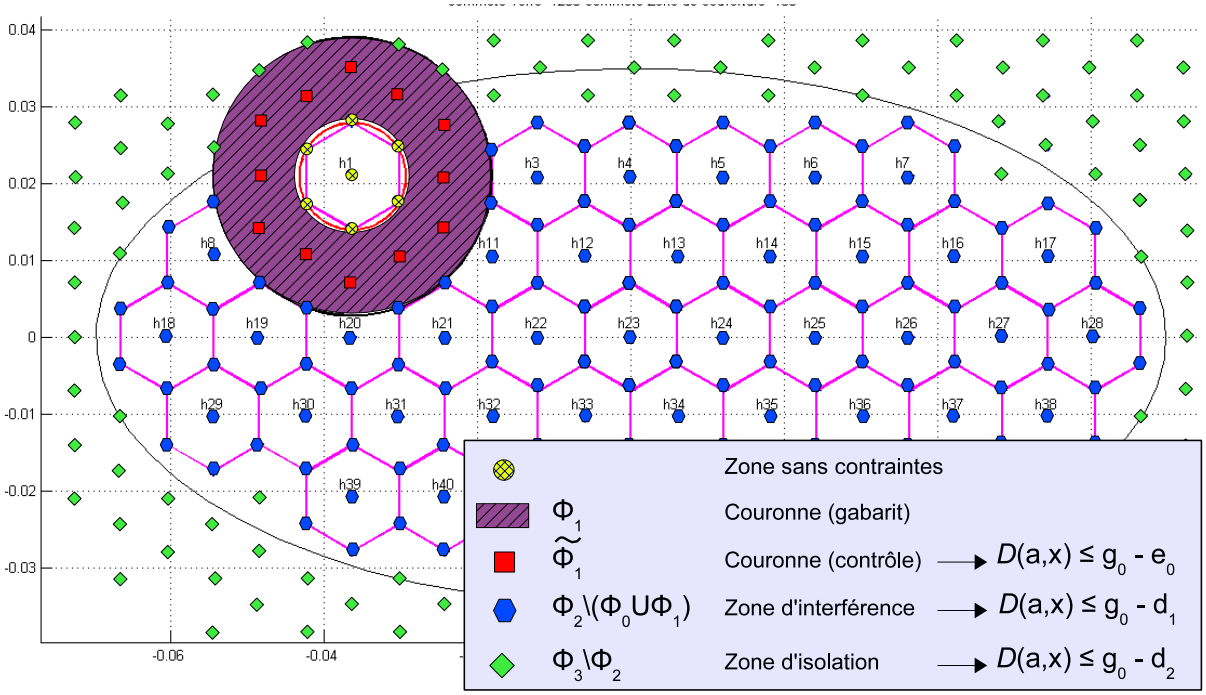


FIG. II.7 – Maillage des domaines pour les nouvelles contraintes

3.3.8 Fonction coût

Evaluation des contraintes par discrétisation On définit une fonction $\widetilde{F} : \mathbb{R}^{2n} \longrightarrow \mathbb{R}^q$ qui évalue les contraintes de gabarit, en utilisant le maillage des points sur la Terre pour discrétiser le problème en x :

$$\widetilde{F}(a) = \begin{pmatrix} \widetilde{F}_1(a) \\ \vdots \\ \widetilde{F}_q(a) \end{pmatrix} = \begin{pmatrix} C_{i_1}(a, x_1)^+ \\ \vdots \\ C_{i_q}(a, x_q)^+ \end{pmatrix} \tag{II.27}$$

avec

- Les q composantes correspondent aux points $\{x_1, \ldots, x_q\}$ du maillage Terre à l'extérieur du spot k.
- Chaque point x_j appartient à un des quatre domaines disjoints Ψ_i associés aux contraintes $C_i,\ i \in \{1, \ldots, 4\}$: on note $i_j \in \{1, \ldots, 4\}$ l'indice correspondant.

  C_{i_j} est la contrainte qu'on veut vérifier au point x_j,
- $x \mapsto x^+ := max(x, 0)$ la fonction **partie positive**.

Ainsi pour chaque composante de $\widetilde{F}$:

- $\widetilde{F}_j > 0$ quand la contrainte C_{i_j} en x_j n'est pas vérifiée,
- $\widetilde{F}_j = 0$ quand la contrainte C_{i_j} en x_j est vérifiée.

Pondération Nous allons affecter un poids $\lambda_j \in \mathbb{R}^+$ différent à chaque contrainte, avec l'intérêt de pouvoir les considérer comme des paramètres de la fonction coût qu'on peut faire varier pour ajuster les résultats :

- Si l'algorithme n'arrive pas à satisfaire certaines contraintes, on peut augmenter les poids correspondants, on augmente alors la contribution de ces contraintes à la fonction coût, et la priorité de l'algorithme sera de minimiser celles-là.
- Dans notre cas, les poids permettent aussi de mettre à la même échelle de valeurs les différentes contraintes.

Exemple En effet, si on écrit les contraintes avant la conversion en décibels, avec par exemple $\widetilde{g_0} = 40$ dB et $\widetilde{d_1} = 27$ dB, on obtient :

- $g_0 = e^{\widetilde{g_0}\frac{\ln 10}{10}} \simeq 10000$,
- $\widetilde{g_0} - \widetilde{d_1} = 13$dB et $g_0 - d_1 = e^{27\frac{\ln 10}{10}} \simeq 20$.

Ainsi au niveau des contraintes :

- $C_1(a, x_j) = \mathcal{D}(a, x_j) - g_0 \simeq \mathcal{D}(a, x_j) - 10000$,
- et $C_2(a, x_j) = \mathcal{D}(a, x_j) - g_0 + d_1 \simeq \mathcal{D}(a, x_j) - 20$.

Donc au cours des itérations, si d_1 et d_2 ne sont pas respectées respectivement en deux points x_1 et x_2, avec par exemple $\mathcal{D}(a, x_1) = \mathcal{D}(a, x_2) = 0$, leurs valeurs présentent une différence de $10000 - 20 \backsimeq 10000$.

Mais on peut mettre la contribution des points x_1 et x_2 à la même échelle en prenant $\lambda_1 = 1$ et $\lambda_2 = 10^4$.

Nous choisirons donc les poids sur les contraintes en suivant la même idée, adaptée aux valeurs de gabarit des cas tests.

Formulation Après pondération des contraintes, nous obtenons

$$F(a) = \begin{pmatrix} F_1(a) \\ \vdots \\ F_q(a) \end{pmatrix} = \begin{pmatrix} \lambda_{i_1}\ \widetilde{F}_1(a) \\ \vdots \\ \lambda_{i_q}\ \widetilde{F}_q(a) \end{pmatrix} \tag{II.28}$$

avec les poids $\lambda_{i_j} \in \mathbb{R}^+$ qui prennent une des quatre valeurs qu'on associe à une des contraintes $C_i,\ i \in \{1, \ldots, 4\}$.

Cette fonction ne dépend plus que de a et permet de contrôler le rayonnement en chaque sommet du maillage Terre hors spot utile : les composantes de F sont toutes nulles quand les contraintes C_{i_j} sont satisfaites pour tous les points x_j.

Fonction à minimiser La fonction coût $J : \mathbb{R}^{2n} \longrightarrow \mathbb{R}^+$ est alors définie par la norme au carré de la fonction précédente :

$$J(a) = \frac{1}{2}\ \|F(a)\|^2 = \frac{1}{2}\ F^T(a)\ F(a). \tag{II.29}$$

Si on l'exprime en fonction des contraintes on obtient

$$\begin{aligned} J(a) &= \tfrac{1}{2}\ \sum_{j=1}^{q} [F_j(a)]^2 \\ &= \tfrac{1}{2}\ \sum_{j=1}^{q} \left[C_{i_j}(a, x_j)^+\right]^2. \end{aligned} \tag{II.30}$$

Ainsi, J est **positive** et **vaut zéro quand les contraintes** c_j sont **satisfaites sur tous les points du maillage** Terre : **minimiser** J revient à chercher **la loi d'alimentation** solution du problème transformé $(\mathcal{Q}_1)$, et il faudra vérifier qu'elle est aussi solution du problème de départ $(\mathcal{P}_0)$.

3.3.9 Algorithme d'optimisation

Nous avons construit J de manière à avoir une fonction coût numériquement différentiable.

De plus la directivité $\mathcal{D}$ qui intervient dans l'expression des composantes de F est connue sous forme analytique : nous pouvons **calculer son gradient par rapport aux parties réelles et imaginaires** des composantes de $a = (x_{a_1}, \ldots, x_{a_n}, \vdots\, y_{a_1}, \ldots, y_{a_n}) \in \mathbb{R}^{2n}$ avec $a_j = x_{a_j} + i\ y_{a_j}$.

Le gradient de J est ainsi disponible pour un coût négligeable, et nous pouvons choisir de minimiser avec une méthode de descente : nous avons choisi un **algorithme de Levenberg-Marquardt** (voir section 3.3.3 du chapitre I page 40).

Le principe est de calculer une direction de descente en approchant la hessienne de F par $DF^T DF$: cela rend la méthode performante car on bénéficie d'une meilleure précision que les méthodes d'ordre un qui n'utilisent que le gradient.

De plus, après avoir reformulé le problème, qui revient à minimiser la directivité de type quadratique en a, nous avons un **problème d'optimisation convexe avec un seul minimum, vers lequel converge une méthode de descente** quel que soit le point de départ.

La méthode donne le système linéaire suivant à résoudre pour trouver la direction de descente d_k (voir section 3.3.3 du chapitre I page 41) :

$$\left(\sum_{j=1}^{p} \nabla f_j(x_k) \, \nabla f_j(x_k)^T \; + \; \rho \; I_n \right) \; d_k = - \; \sum_{j=1}^{p} f_j(x_k) \; \nabla f_j(x_k). \tag{II.31}$$

L'algorithme utilisé pour calculer une loi d'alimentation optimale a est présenté dans le tableau II.3.

Nous appliquons une loi physiquement réalisable en normalisant le résultat $a_{opt} := \frac{a_{opt}}{\|a_{opt}\|}$ (voir explication en section 3.2.2 page 62).

3.3.10 Affichages

Nous détaillons ici les affichages qui apparaissent dans les résultats des tests numériques.

Cartes de module et phase L'alimentation optimale de l'antenne est calculée sous la forme partie réelle et partie imaginaire, mais elle est donnée en sortie sous forme d'un vecteur de $\mathbb{C}^n$.

Or un moyen simple de visualiser la loi obtenue est de représenter pour chaque ERel d'une part le module, d'autre part la phase.

Les affichages sont obtenus à l'aide d'une échelle de couleurs, voir par exemple la figure II.19 page 83 :

- Pour les modules, l'échelle varie entre 0 et 0.1 (pour indication, pour une antenne avec 500 ERel, la valeur cible est $\frac{1}{\sqrt{500}} \simeq 0.044$, voir section 4.2.3 page 89),
- Pour les phases, l'échelle varie entre $-\pi$ et π.

  Nous donnons la même couleur à $-\pi$ et à π, car la phase est définie modulo 2π.

Fonction coût Nous affichons la fonction coût normalisée avec la valeur de départ $\frac{J_k}{J_0}$ en fonction des itérations, pour pouvoir comparer des pondérations différentes des composantes de F : voir par exemple la figure II.13 page 80.

Niveaux de directivité Pour une alimentation donnée, nous affichons les niveaux de directivité de l'antenne sur le maillage de la Terre avec des courbes d'isovaleurs : voir par exemple la figure II.26 page 99.

Cela permet de visualiser la largeur et la régularité du lobe principal, ou les remontées des lobes de réseau.

Algorithme 1 : Optimisation d'une loi d'alimentation

▷ **Initialisation**

- ε précision souhaitée,
- $a_k = a_0$ point de départ,
- it_M nombre maximal d'itérations,

▷ **Mise à jour des variables**

- $F_k = F(a_k)$
- $J_k = \frac{1}{2} F_k^T F_k$
- $DJ_k = DF(a_k)^T F(a_k)$
- $M_k = DF(a_k)^T DF(a_k) + \rho\ I_n$

▷ **Boucle d'optimisation**

- $it = 0$
- $J_0 = J_k$
- Tant que $(it\ <\ it_M)$ et $(\frac{J_k}{J_0} > \varepsilon)$
  - résolution de $M_k\ d_k = -DJ_k$ par la méthode du gradient conjugué
  - t_k obtenu par recherche linéaire
  - $a_k := a_k + t_k\ d_k$
  - mise à jour des variables
  - $it = it + 1$

▷ **Sortie**

- $a_{opt} := a_k$ alimentation optimale

TAB. II.3 – Algorithme d'optimisation d'une loi d'alimentation

Les valeurs d'iso-niveaux correspondant aux seuils de directivité à respecter sont marquées sur la figure, par exemple pour le premier cas test :

- 40 dB pour les points de la zone utile,
- 13 dB pour les points de la zone d'interférence,
- 20 dB pour les points de la zone d'isolation.

Remarque La grille utilisée pour tracer les courbes d'iso-niveaux est générée par le logiciel (on peut calculer la directivité en un point quelconque grâce à la formule analytique) et est différente du maillage utilisé par l'algorithme. Il peut y avoir des zones où le rayonnement remonte au-dessus des valeurs données par la sortie de l'algorithme, sur des points extérieurs au maillage, mais l'affichage reste pertinent pour visualiser le résultat.

Performances de l'antenne Les lois d'alimentation sont calculées pour l'ensemble des spots de la zone de couverture, on affiche alors un graphique permettant de visualiser si les contraintes de gabarit sont respectées pour l'ensemble des spots et éventuellement pour lesquels l'algorithme est défaillant : voir par exemple la figure II.12 page 79.

Les trois valeurs de gabarit à respecter sont représentées par trois droites horizontales et on affiche pour chaque spot trois valeurs de directivité en décibels :

- la valeur minimum sur zone utile,
- la valeur maximum sur zone d'interférence,
- la valeur maximum sur zone d'isolation.

3.4 Résultats numériques

3.4.1 Paramètres d'entrée

Dans le cadre de cette étude, on dispose des paramètres d'entrée suivants :

Antenne de départ

- nombre d'ERel $n = 440$,
- $P_j = Q_j = 4$ (ER 4×4 patches).

Directivité

- longueur d'onde $\lambda = 15\text{mm}$,
- $\alpha = 3.48023$ et $\theta_1 = \frac{2\pi}{3}$.

Zones sur la Terre

- rayon de la Terre $\mathcal{R}_t = 9^\circ$,
- zone de couverture elliptique $\mathcal{R}_u = 4^\circ$ et $\mathcal{R}_v = 2^\circ$,
- rayon d'un spot $\mathcal{R}_{sp} = 0.4^\circ$,
- rayon d'une couronne $\mathcal{R}_c = \frac{3\sqrt{3}}{2}\mathcal{R}_{sp}$,
- test "Centre" : $(U_1, V_1) = (0, 0)$ le centre de la Terre.

Les dimensions précédentes, en coordonnées sphériques, sont converties en coordonnées cartésiennes pour être utilisées dans les calculs.
On note par exemple $R_t = sin\left(\frac{\mathcal{R}_t \pi}{180}\right)$.

La zone de couverture est constituée de 45 spots adjacents (voir figure II.7).

Gabarit

- $\widetilde{g_0} = 40$ dB valeur minimum sur spot utile,
- niveaux de recoupement
  - $\widetilde{d_0} = 4$ dB
  - $\widetilde{d_1} = 27$ dB donc $\widetilde{g_0} - \widetilde{d_1} = 13$ dB valeur minimum sur la zone d'interférence,
  - $\widetilde{d_2} = 20$ dB donc $\widetilde{g_0} - \widetilde{d_2} = 20$ dB valeur minimum sur la zone d'isolation.

3.4.2 Modélisation

Les paramètres d'entrée sont utilisés avec les données de la section 2.1 : on utilise dans ce cas test uniquement des formules pour modéliser le problème.

Pour pouvoir développer rapidement un code opérationnel, nous avons utilisé le langage de programmation du logiciel MATLAB.

Antenne L'antenne est représentée par une matrice

$$Mer = \begin{pmatrix} x_1^1 & \dots & x_n^1 \\ y_1^1 & \dots & x_n^1 \\ x_1^2 & \dots & x_n^2 \\ y_1^2 & \dots & x_n^2 \end{pmatrix} \in \mathbb{R}^{4\times n} \tag{II.32}$$

avec

- (x_i^1, y_i^1) premier coin (en haut à gauche) de l'ER de centre M_i,
- (x_i^2, y_i^2) quatrième coin (en bas à droite) de l'ER.

Les coordonnées sont calculées de telle manière que les 440 ER remplissent la surface de l'antenne de diamètre 1.35 m : voir figure II.8.

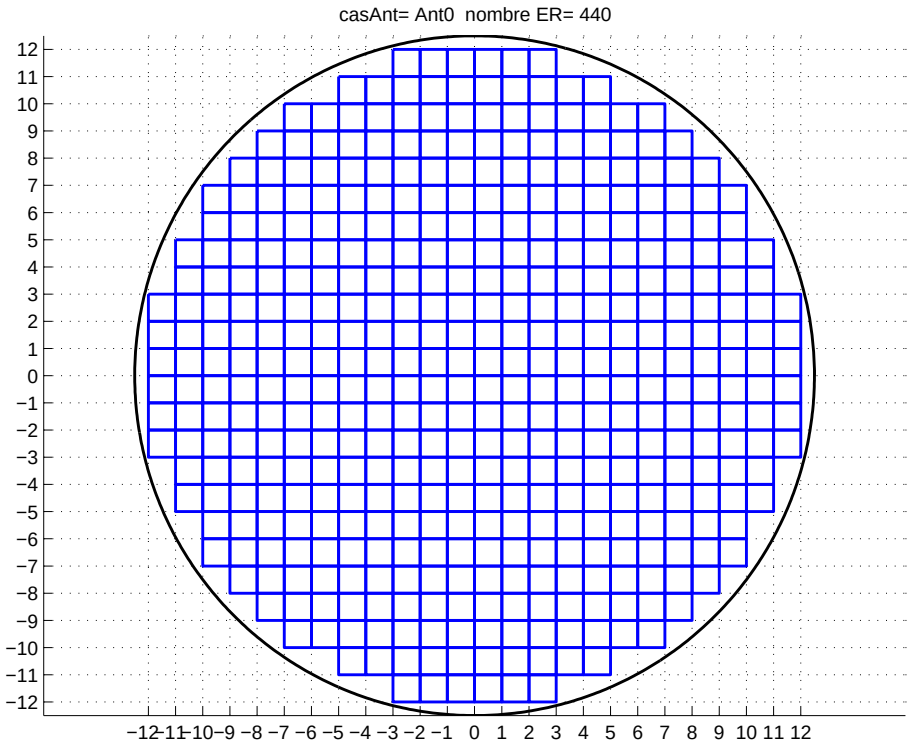


FIG. II.8 – Antenne de départ : 440 ER

Maillage Terre Le maillage des zones sur la Terre ne fait pas partie des spécifications (voir figures II.9 et II.10) :

- Nous avons choisi de modéliser la Terre par un maillage à motif hexagonal, ceci pour pouvoir mailler et contrôler le mieux possible le bord des spots circulaires avec le moins de points possible,
- Nous définissons le pas du maillage de la zone de couverture afin que les sommets d'un hexagone soient situés sur le bord d'un spot, et nous maillons un peu au delà de la zone de couverture pour avoir des points de contrôle également sur les couronnes des spots situés au bord.
- Le pas du maillage hors zone de couverture est plus grand que le précédent pour ne pas avoir trop de points au total.

  Le maillage hors zone de couverture est une extension du précédent maillage avec des hexagones trois fois plus grands.
- On obtient $p \simeq 1200$ points de maillage.

Ce maillage est représenté par une matrice

$$M_{esp} = \begin{pmatrix} u_1 & \dots & u_p \\ v_1 & \dots & v_p \end{pmatrix} \in \mathbb{R}^{2\times p}. \tag{II.33}$$

Dans ce test, la zone de couverture est centrée au centre de la Terre (voir figure II.9).

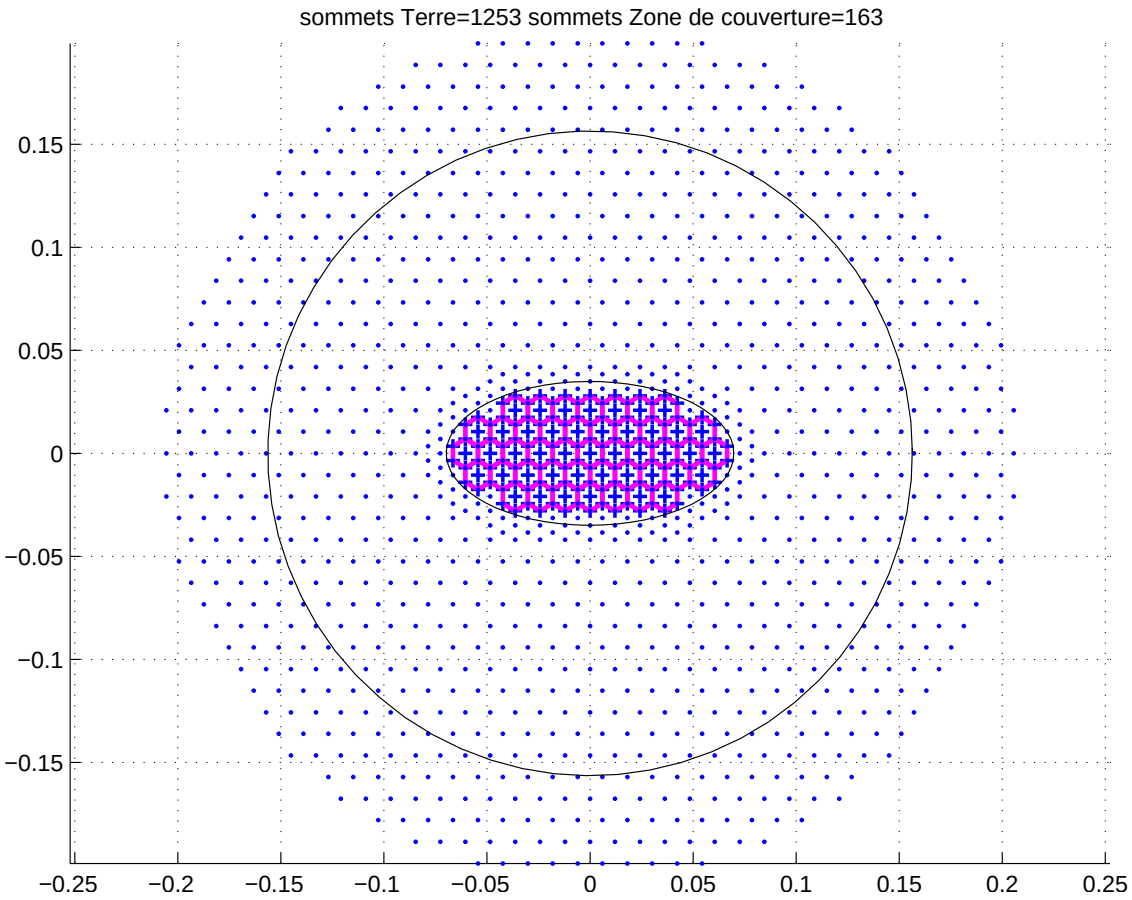


FIG. II.9 – Maillage Terre

Performances On lance l'algorithme en prenant successivement comme spot utile chacun des 45 spots.

On utilise les paramètres suivants :

- poids fixes sur les contraintes : $\lambda_1 = 1$, $\lambda_2 = 10^2$, $\lambda_3 = 10^2$, $\lambda_4 = 10^3$,
- e_0 telle que sa valeur exprimée en dB soit $\widetilde{e_0} = 3$ dB,
- nombre maximum d'itérations : $it_M = 20$,
- alimentation initiale a_0 vecteur constant : elle produit un lobe principal dans la direction où pointe l'antenne, c'est-à-dire au centre de la zone de couverture (voir figure II.11).

Les performances sur tous les spots sont présentées dans la figure II.12 .

Analyse Nous obtenons les résultats suivants :

- L'algorithme a réussi à faire descendre pour la plupart des spots la fonction coût jusqu'à zéro.

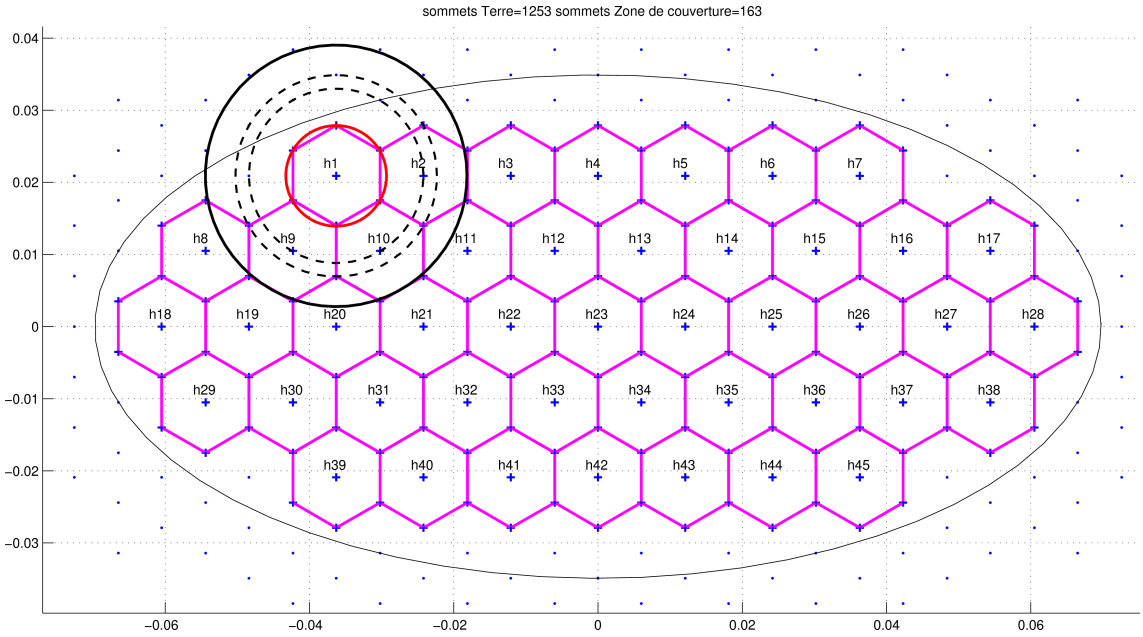


FIG. II.10 – Maillage zone de couverture

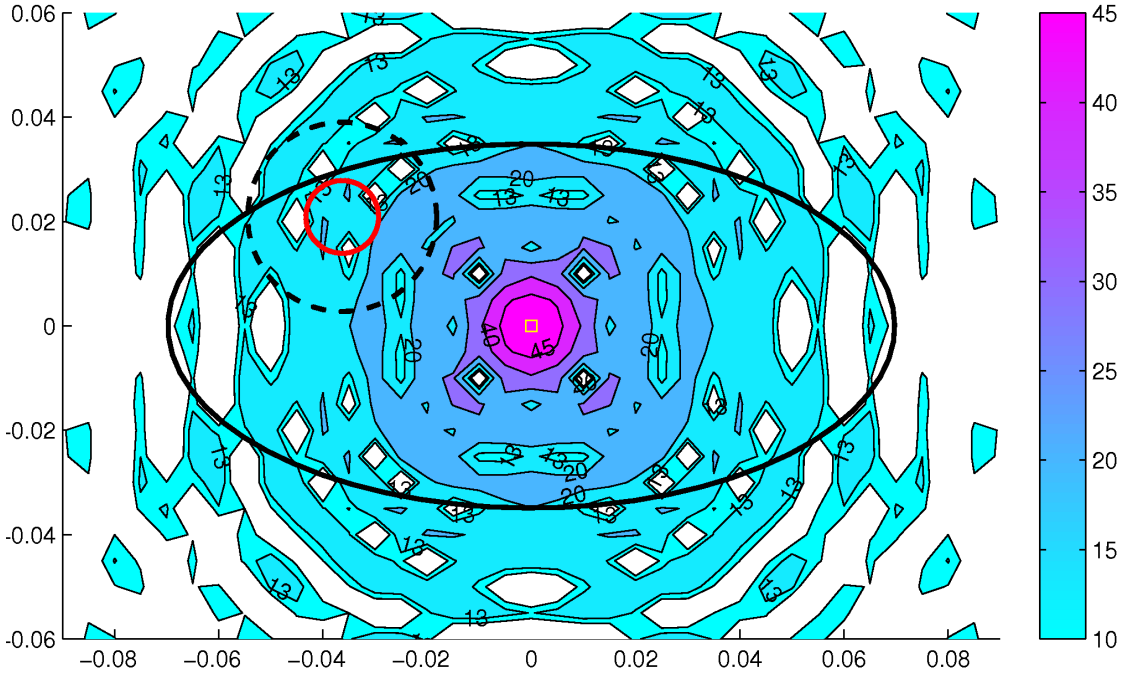


FIG. II.11 – Spot n°1 : courbes iso-niveaux de la directivité pour l'alimentation initiale

Les contraintes spécifiées sont vérifiées exactement et l'algorithme s'arrête avant le nombre maximal d'itérations (20) : l'algorithme d'optimisation des alimentations est performant et robuste.

- Les critères ne sont pas satisfaits pour deux spots : n°18 et 28.

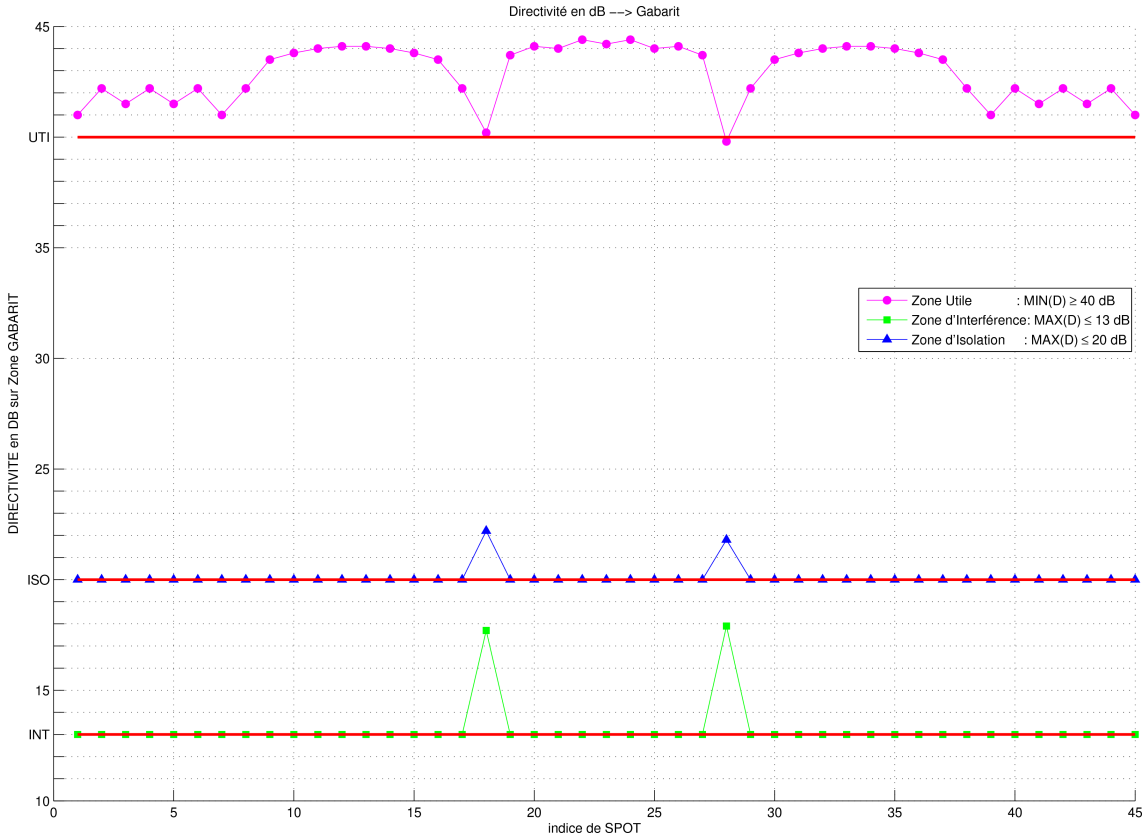


FIG. II.12 – Performances : antenne de départ

Obervons les résultats sur un exemple où le gabarit est satisfait, nous présentons pour le spot n°1

- en fonction des itérations dans la figure II.13
  - la valeur de la fonction coût normalisée $\frac{J_k}{J_0}$,
  - les niveaux de directivité D sur les zones de gabarit,
- en ce qui concerne l'alimentation optimale
  - dans la figure II.14 les niveaux de rayonnement,
  - dans la figure II.15 les cartes de modules et phases.

On observe que la fonction coût diminue régulièrement :

- après 4 itérations, le lobe principal est dépointé dans le spot utile (la directivité est supérieure à 40 dB dans la zone de gain),
- les itérations suivantes servent à abaisser les lobes secondaires sur les zones d'interférence et d'isolation.

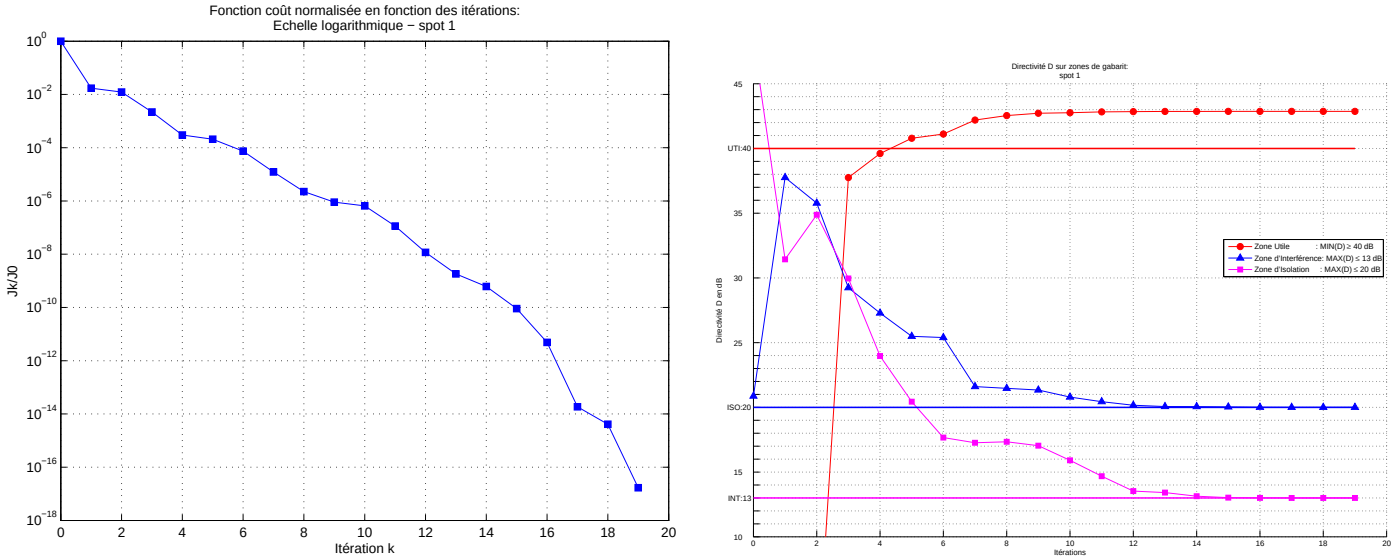


FIG. II.13 – Spot n°1
A gauche : fonction coût normalisée $\frac{J_k}{J_0}$ en fonction des itérations
A droite : directivité D (dB) sur zones de gabarit en fonction des itérations

Pondération initiale	Pondération améliorée
$\lambda_1 = 1$	$\lambda_1 = 1$
$\lambda_2 = 10^2$	$\lambda_2 = 10^4$
$\lambda_3 = 10^2$	$\lambda_3 = 10^3$
$\lambda_4 = 10^3$	$\lambda_4 = 10^2$

TAB. II.4 – Pondération des contraintes

Pondération améliorée Nous allons essayer d'améliorer les résultats sur les deux spots pour lesquels les critères ne sont pas satisfaits : ils sont situés aux deux extrémités horizontales de la zone de couverture. Ils correspondent aux cas où on doit dépointer le plus le faisceau par rapport au centre de la zone de couverture.

Nous observons les résultats pour le spot 18 avec la pondération initiale sur les figures associées.

Nous voyons sur la figure II.16 de gauche que la fonction coût diminue au cours des cinq premières itérations, quand le lobe principal est dépointé du centre de la zone de couverture vers le spot, ce que montre la courbe de directivité sur la zone utile sur la figure II.17 de gauche. Puis la fonction coût ne diminue plus, ce qui traduit la difficulté de l'algorithme à abaisser le rayonnement, surtout dans la zone d'interférence comme le montre la courbe de directivité sur la zone d'interférence sur la figure II.17 de gauche, qui présente plusieurs phases de croissance.

Nous allons agir sur la pondération des contraintes en augmentant fortement le poids λ_2 associé à la contrainte sur la zone d'interférence et en augmentant aussi le poids λ_3 associé à la contrainte sur la zone d'isolation (voir tableau II.4).

Nous présentons pour le spot n°18 en fonction des itérations et pour les deux pondérations

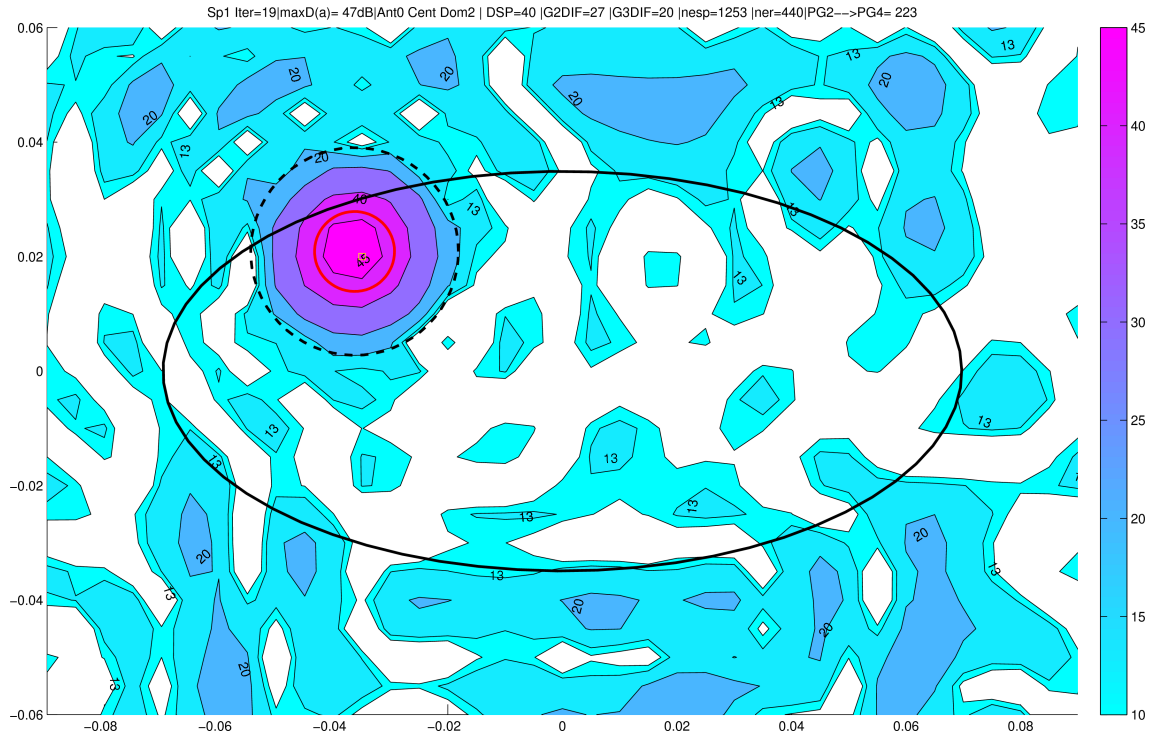


FIG. II.14 – Spot n°1 : courbes iso-niveaux de la directivité pour l'alimentation optimale

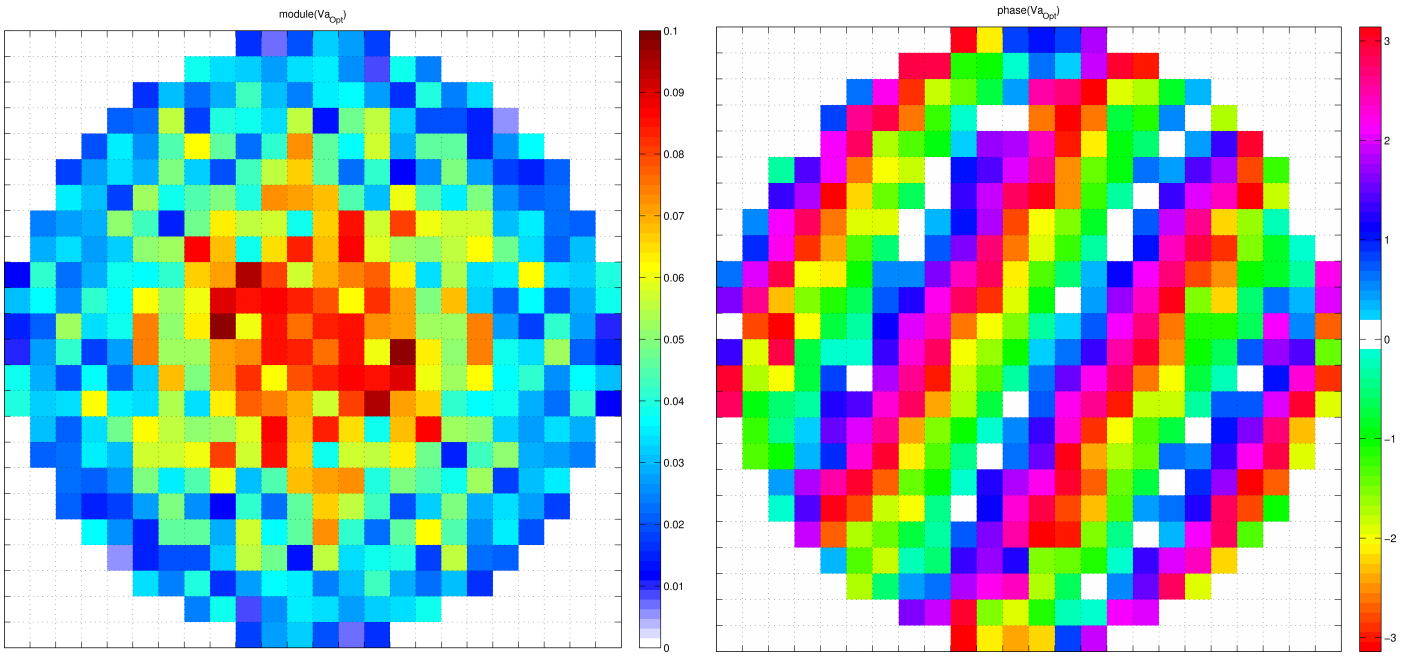


FIG. II.15 – Spot n°1 : alimentation optimale
A gauche : carte des modules
A droite : carte des phases

- la valeur de la fonction coût normalisée $\frac{J_k}{J_0}$ dans la figure II.16,

▪ les niveaux de directivité D sur les zones de gabarit dans la figure II.17.

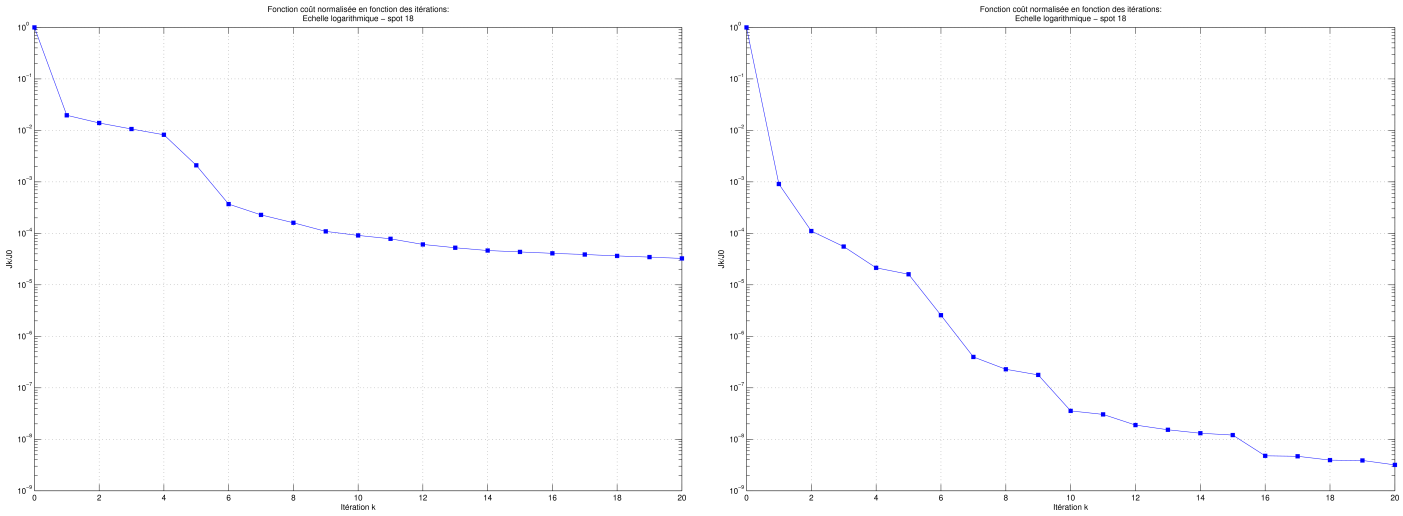


FIG. II.16 – Spot 18 : fonction coût normalisée $\frac{J_k}{J_0}$ en fonction des itérations
A gauche : pondération initiale
A droite : pondération améliorée

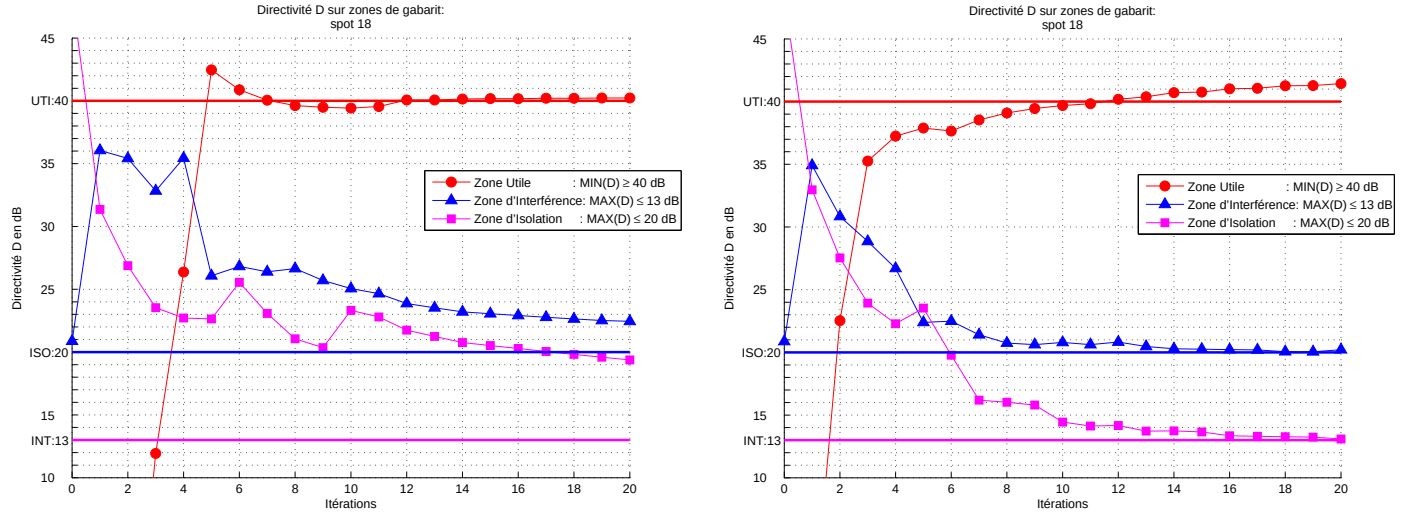


FIG. II.17 – Spot 18 : directivité D (dB) sur zones de gabarit en fonction des itérations
A gauche : pondération initiale
A droite : pondération améliorée

Avec la deuxième pondération, la fonction coût diminue tout au long des itérations, et les trois courbes de directivité convergent vers les valeurs demandées.

On observe dans la figure II.18 des courbes iso-niveaux de directivité que le rayonnement est mieux concentré dans le spot avec la deuxième pondération, mais l'énergie maximale obtenue dans le spot est inférieure à celle obtenue dans le cas du spot 1, car ici on dépointe le signal plus loin.

Nous présentons à présent pour le spot n°18 à propos de l'alimentation optimale et pour les deux pondérations

- dans la figure II.18 les niveaux de rayonnement,
- dans les figures II.19 et II.20 les cartes de modules et phases.

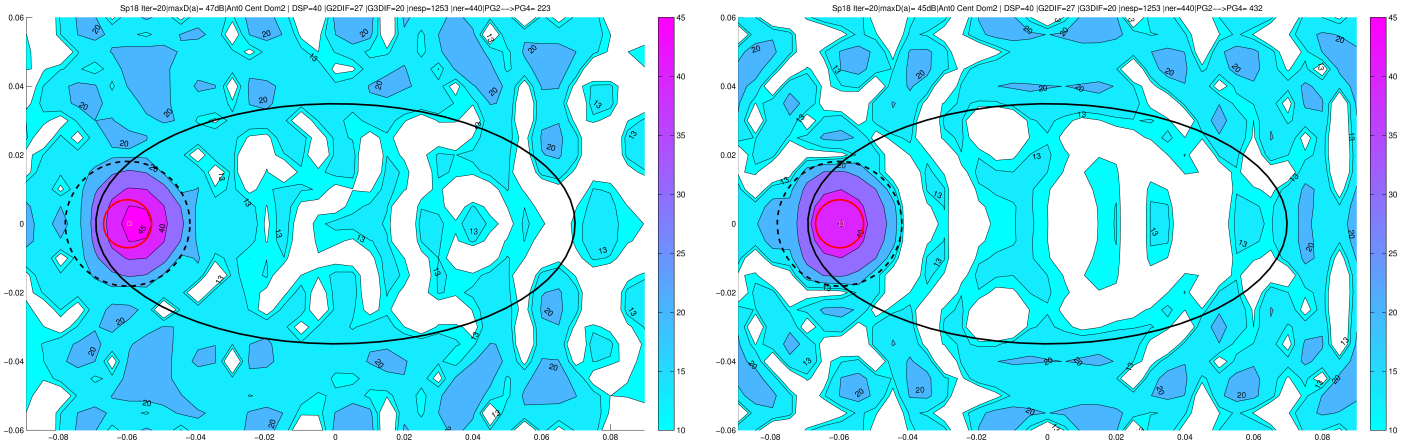


FIG. II.18 – Spot 18 : courbes iso-niveaux de la directivité pour l'alimentation optimale
A gauche : pondération initiale
A droite : pondération améliorée

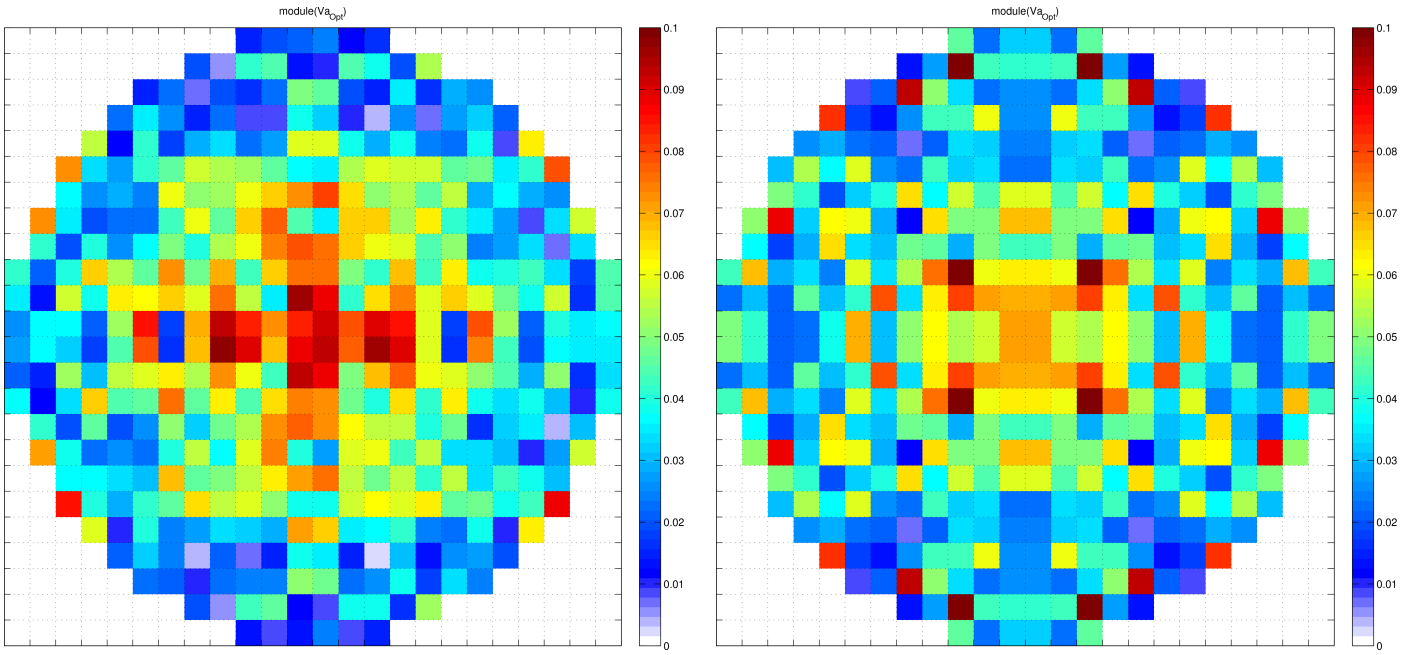


FIG. II.19 – Spot n°18 : alimentation optimale
A gauche : carte des modules avec la pondération initiale
A droite : carte des modules avec la pondération améliorée

Nous constatons que le rayonnement est mieux concentré dans le spot, et cela se traduit par une carte des modules plus symétrique et une carte des phases avec des bandes plus régulières.

Nous obtenons des résultats similaires pour le spot n° 28.

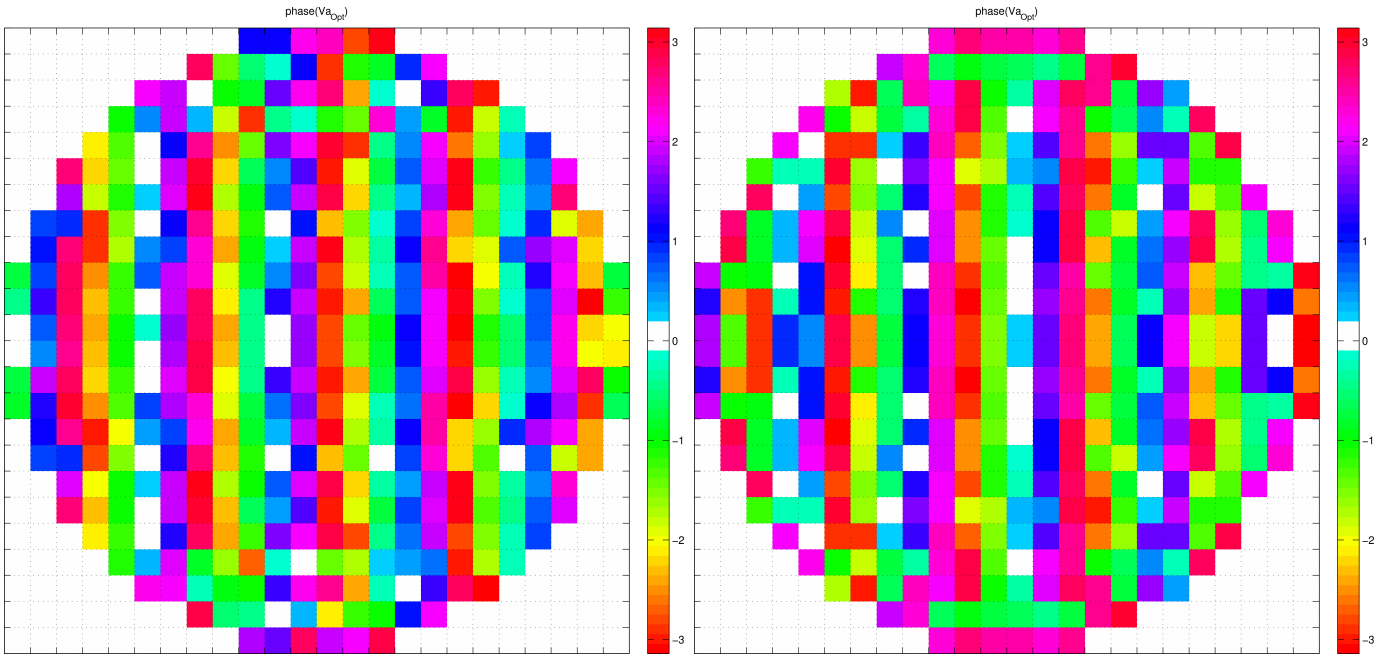


FIG. II.20 – Spot n°18 : alimentation optimale
A gauche : carte des phases avec la pondération initiale
A droite : carte des phases avec la pondération améliorée

Conclusion En modifiant la pondération des contraintes pour deux spots, nous obtenons finalement les lois d'alimentation optimales qui vérifient le gabarit spécifié pour tous les spots.

Ce cas test nous a permis de tester la validité de l'algorithme de calcul des lois optimales : il s'est avéré performant.

Nous avons aussi commencé à mettre en place la méthode de regroupement d'ER qui ne constituait pas encore un algorithme (une antenne optimale a été construite manuellement, ce résultat n'est pas présenté ici car la méthode n'était pas automatique).

4 Etude industrielle : projet OTOP

4.1 Introduction

L'étude de faisabilité précédente ayant donné des résultats encourageants, Thalès a fourni ensuite au laboratoire MIP un cas test industriel dans le cadre du projet OTOP (pour Optimisation Topologique des équipements-clé des télécommunications du futur) : coordonné par le laboratoire MIP, ce projet a été retenu par l'ANR-RNRT.

L'objectif est de trouver une méthode pour calculer la géométrie du réseau utilisant le moins possible d'ER tout en continuant à vérifier les contraintes de gabarit.

Il faut un algorithme robuste et une méthode entièrement automatisée, car dans le cadre du projet il faut livrer un logiciel pour un utilisateur non expert en optimisation.

Dans cette étude, la directivité est donnée par une formule explicite quadratique par rapport à l'alimentation a, mais pour des raisons industrielles, on souhaite ici **contrôler uniquement la phase** : les modules de a sont fixés.

Dans le problème de recherche des phases optimales de $a \in \mathbb{C}^n$:

- la dimension de l'espace de travail est n,
- il n'y a pas de contrainte sur a.

Cependant, notre approche est différente :

- Nous préférons utiliser comme **variables de calcul** les **parties réelle et imaginaire** des composantes de a pour **réduire la non-linéarité** introduite par la phase, quitte
  - à travailler en dimension $2n$,
  - et à avoir une contrainte supplémentaire sur a pour respecter les valeurs fixes de modules.
- De plus, dans l'objectif de disposer d'un **critère pour effectuer les regroupements d'ER** dans l'**optimisation topologique**, nous aurons besoin de récupérer le **degré de liberté sur les modules**, même si cela ne correspond pas à l'alimentation souhaitée (cas dit "relaxé").

Pour **reformuler le problème sous forme convexe** comme précédemment :

- Nous allons ici transformer la **contrainte d'égalité sur les modules en contrainte d'inégalité** pour rendre l'ensemble admissible convexe (à l'optimum les contraintes saturées vérifierons l'égalité demandée),
- Les zones de contrôle du rayonnement ont changé par rapport à l'étude préliminaire, nous adaptons l'idée issue de la conservation de l'énergie :

  au lieu de maximiser le rayonnement à l'intérieur de la zone utile, nous allons **imposer une valeur de champ au centre du spot** et obtenir un problème de minimisation convexe.

Nous présentons ensuite les performances de l'antenne de départ avec les lois optimales, pour l'ensemble des spots utiles de la zone de couverture, dans le cas relaxé des contraintes sur les modules : celui dont nous avons besoin pour l'optimisation topologique.

Les lois optimales dans le cas non relaxé que l'on veut appliquer seront calculées avec l'antenne finale, optimisée par regroupements d'ER.

4.2 Données

4.2.1 Introduction

Les données de ce cas test de type industriel ont été générées par le logiciel commercial de calcul du rayonnement GRASP (développé par la société Ticra), utilisé par Thalès. Ce logiciel sert à modéliser des antennes réseaux, à calculer leurs lois d'alimentation optimales, mais ne permet pas de créer une antenne optimisée, ni ce calculer les lois optimales pour une antenne à réseau irrégulier.

Les données ont été fournies sous forme de fichiers au format standardisé pour le logiciel GRASP.

Les hypothèses sont en partie les mêmes que celles présentées en section 2.1 pour l'étude préliminaire, nous présentons dans la suite les différences apportées dans cette deuxième étude.

4.2.2 Paramètres d'entrée

Nous avons codé des procédures pour pouvoir lire les fichiers de données fournis par Thalès et les transformer en matrices (variables MATLAB).

Antenne de départ Ant_0 La description de l'antenne de départ, notée Ant_0, est fournie dans un fichier ".isp" qui contient les coordonnées des ER :

- $n_0 = 529$ ERel (voir figure II.21),
- chaque ERel est un carré de 54 mm.

Zones sur la Terre Le maillage de la Terre est fourni dans un fichier ".sta" qui contient les coordonnées de p=5748 points.

Contrairement à l'étude préliminaire, on ne dispose pas de points de maillage dans l'espace hors Terre (voir figure II.22).

La zone de couverture est composée de 44 spots de taille 0.8°.

Gabarit Ce fichier contient aussi les valeurs de gabarit données en décibels :

- $\widetilde{g_0} = 42$ dB valeur minimum sur spot utile
- niveaux de recoupement

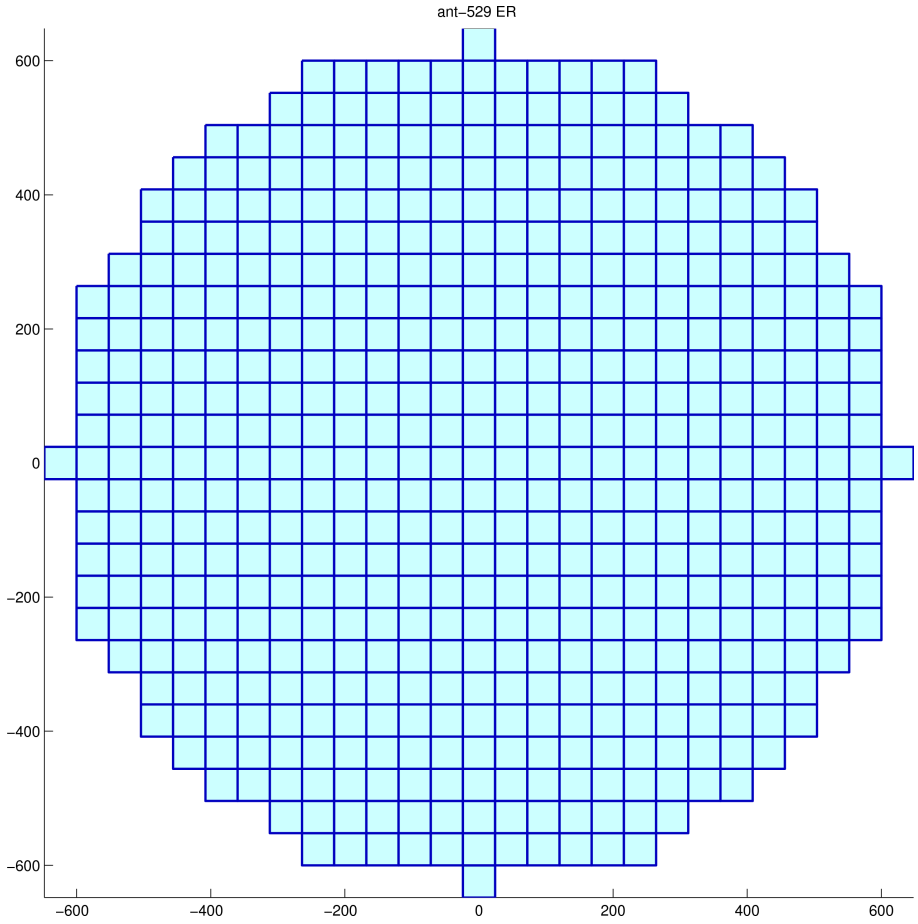


FIG. II.21 – Antenne de départ : 529 ER

- $\widetilde{d_1} = 25$ dB donc $\widetilde{g_0} - \widetilde{d_1} = 17$ dB valeur maximum pour la zone d'interférence,
- $\widetilde{d_2} = 20$ dB donc $\widetilde{g_0} - \widetilde{d_2} = 22$ dB valeur maximum pour la zone d'isolation.

Champ d'un ERel Le champ a été calculé par le logiciel de modélisation de Thalès avec l'antenne précédente, et les valeurs de champ notées $E_j(x_k)$ sont fournies dans un fichier ".grd" pour chaque ERel d'indice j et en chaque point du maillage Terre x_k.

Champ du réseau On obtient le champ du réseau pour une alimentation a en un point x_k par produit scalaire du vecteur a avec le vecteur des champs d'ERel en x_k

$$E(a, x_k) = \sum_{j=1}^{n_0} a_j E_j(x_k). \tag{II.34}$$

Normalisation en puissance Les valeurs de champ des ERel fournies dans le fichier sont normalisées pour obtenir une énergie rayonnée totale égale à $1W$, c'est-

à-dire

$$\sum_{k=1}^{p} |E(a, x_k)|^2 = 1. \tag{II.35}$$

Directivité : propriété quadratique convexe La directivité $\mathcal{D}$ pour una alimentation a en un point x_k s'obtient avec la formule donnée en section 3.2.2 page 61 dans l'étude préliminaire. On obtient ici d'après la normalisation en puissance précédente :

$$\mathcal{D}(a, x_k) = |E(a, x_k)|^2 = \left| \sum_{j=1}^{n_0} a_j E_j(x_k) \right|^2. \tag{II.36}$$

La propriété quadratique convexe de la directivité par rapport aux composantes en partie réelle et partié imaginaire de a est conservée.

Zones sur la Terre On reprend les mêmes notations que celles de la section 3.2.3 dans l'étude préliminaire : voir les tableaux II.1 et II.2 pages 63 et 63.

Les différences avec l'étude précédente sont les suivantes :

- la couronne autour du spot n'est pas définie, car ici nous travaillons à une fréquence donnée, et les spots de la zone de couverture associés ne sont plus adjacents (voir figure II.23 page 93),
- la zone hors Terre n'est pas définie, et la contrainte de pertes par débordement n'existe plus (voir figure II.22).

Les zones sont définies par le fichier des points de maillage Terre (voir figure II.22) et par le fichier de gabarit pour leurs caractéristiques (zones de gain ou d'isolation).

4.2.3 Contraintes

Contraintes de rayonnement Quatre fréquences F_i sont données, et on associe à chacune un sous-ensemble disjoint parmi les s spots qui définissent la zone de couverture.

Quand on choisit un spot k noté Φ_0 comme zone utile, un sous-ensemble de spots utilisant la même fréquence lui est associé. La zone d'interférence toujours notée Φ_2 est le sous-ensemble de spots moins Φ_0 (voir figure II.23 page 93).

Pour un spot Φ_0 fixé, le gabarit est du type de celui présenté en 3.2.4 page 62 dans l'étude préliminaire (sans la contrainte globale hors Terre) :

$$\begin{cases} \forall x_j \in \Phi_0 & \widetilde{g_0} \leqslant \widetilde{\mathcal{D}}(a, x) \leqslant \widetilde{g_0} + \widetilde{d_0} \\ \forall x_j \in \Phi_2 \setminus \Phi_0 & \widetilde{\mathcal{D}}(a, x) \leqslant \widetilde{g_0} - \widetilde{d_1} \\ \forall x_j \in \Phi_3 \setminus \Phi_2 & \widetilde{\mathcal{D}}(a, x) \leqslant \widetilde{g_0} - \widetilde{d_2}. \end{cases} \tag{II.37}$$

Les zones de contraintes sont disjointes, ainsi à tout point x de la Terre considéré correspond une seule contrainte possible.

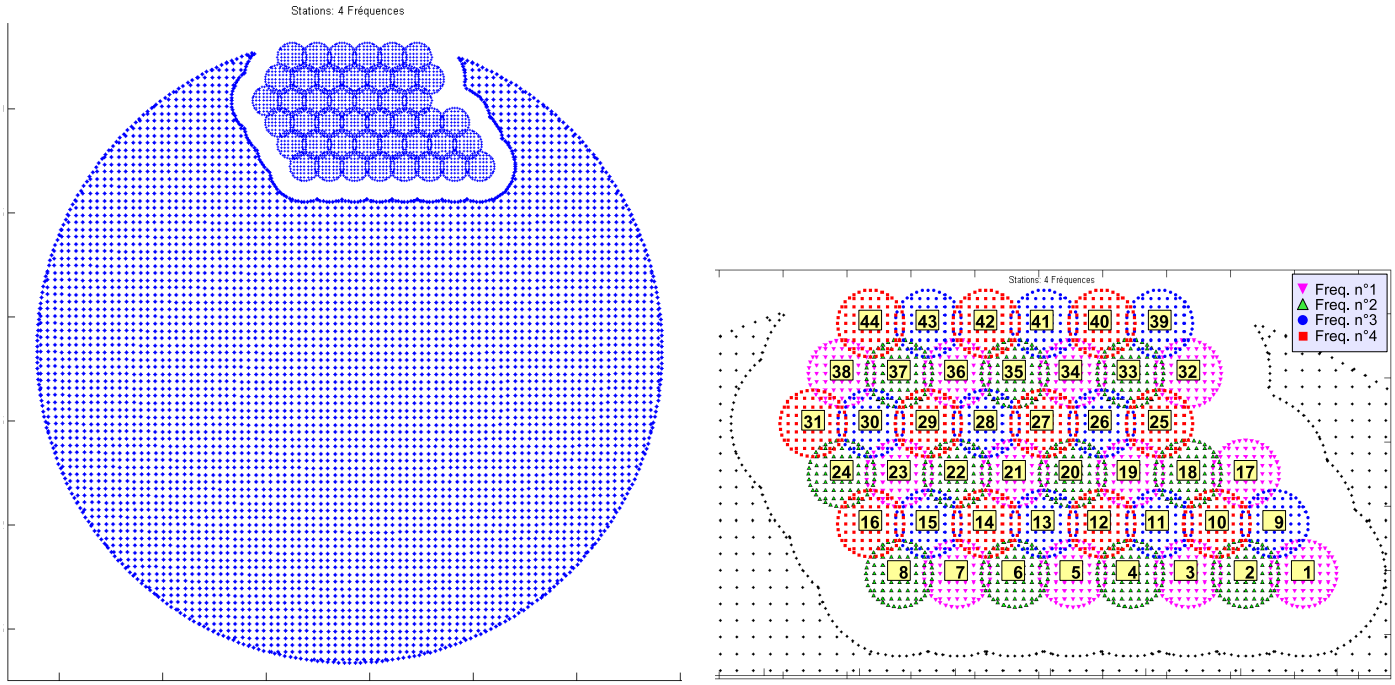


FIG. II.22 – Maillage sur la Terre (à droite quatre motifs pour quatre fréquences)

Contraintes de modules : valeurs cibles Dans cette étude, on impose une nouvelle contrainte : la puissance d'alimentation totale est égale à un

Cela se traduit par la contrainte sur l'alimentation globale :

$$\|a\|^2 = \sum_{j=1}^{n_0} |a_j|^2 = 1. \tag{II.38}$$

Cette puissance est distribuée uniformément sur chaque dispositif électronique de contrôle de l'alimentation : il est en effet plus économique de fabriquer des dispositifs conçus pour contrôler uniquement la phase.

On obtient la contrainte suivante sur les alimentations des ERel :

$$\begin{aligned} & \forall j \in \{1, \dots, n_0\} \quad |a_j|^2 = \tfrac{1}{n_0} \\ \Leftrightarrow \quad & \forall j \in \{1, \dots, n_0\} \quad |a_j| = \tfrac{1}{\sqrt{n_0}}. \end{aligned} \tag{II.39}$$

Dans la suite, on appellera **valeurs cibles** les valeurs que doivent respecter les modules : ici $\frac{1}{\sqrt{n_0}}$ pour chaque ERel.

4.3 Modélisation et idées clés

Nous allons procéder de manière analogue à l'étude préliminaire : nous voulons reformuler le problème afin de rendre la fonction coût le plus régulière possible, ce qui permet d'éliminer les minima locaux et d'utiliser une méthode de descente (l'algorithme de Levenberg-Marquardt) pour trouver le minimum global.

Les contraintes de rayonnement sont similaires à celles de l'étude précédente, mais ici nous ne pouvons pas utiliser le principe de conservation de l'énergie de la même manière.

Idée clé : réduire la non linéarité Nous reprenons l'idée de réduire la non linéarité du problème expliquée dans la section 3.3.2 page 66 de l'étude préliminaire.

Nous utilisons donc pour les **calculs d'optimisation**

- les valeurs de directivité **avant la conversion en décibels**,
- l'alimentation $a \in \mathbb{R}^{2\ n_0}$ en **partie réelle et partie imaginaire**.

4.3.1 Formulation des contraintes

Pour une antenne à géométrie donnée (n ER) et pour un spot k fixé, le problème à résoudre est

$(\mathcal{P}_0)$ trouver $a \in \mathbb{C}^{n_0}$ tel que

$$(\mathcal{P}_0) \quad \begin{cases} \forall x \in \Phi_0 & c_1(a,x) & = -\mathcal{D}(a,x) + g_0 & \leqslant 0 \\ \forall x \in \Phi_0 & c_2(a,x) & = \mathcal{D}(a,x) - g_0 - d_0 & \leqslant 0 \\ \forall x \in \Phi_2 \setminus \Phi_0 & c_3(a,x) & = \mathcal{D}(a,x) - g_0 + d_1 & \leqslant 0 \\ \forall x \in \Phi_3 \setminus \Phi_2 & c_4(a,x) & = \mathcal{D}(a,x) - g_0 + d_2 & \leqslant 0 \\ \\ \forall j \in \{1,\dots,n_0\} & \widehat{c_5}(a_j) & = |a_j|^2 - \frac{1}{n_0} & = 0. \end{cases} \tag{II.40}$$

Il faut vérifier ici deux types de contraintes :

- les contraintes de type inégalité sur la directivité $\mathcal{D}$, ou gabarit,
- les contraintes de type égalité sur les modules de l'alimentation complexe a.

4.3.2 Reformulation : maximisation sous contraintes

Comme en section 3.3.3 page 67, la contrainte c_2 n'est pas indispensable, et la contrainte c_1 qui consiste à maximiser l'énergie $\mathcal{D}$ à l'intérieur du spot va définir le problème de maximisation. La contrainte $\widehat{c_5}$ définit l'ensemble admissible pour la variable d'optimisation a.

On rappelle que pour un point x donné, on note $c_{i_x}(a,x)$ la contrainte associée parmi les cinq précédentes, selon la zone à laquelle appartient x.

Le problème $(\mathcal{P}_0)$ se reformule sous la forme du problème d'optimisation sous contraintes $(\mathcal{P}_1)$:

$$(\mathcal{P}_1) \quad \begin{cases} \max\limits_{a \in \widehat{\mathcal{A}}} \ \min\limits_{x \in \Phi_0} \ \mathcal{D}(a,x) \\ \\ \widehat{\mathcal{A}} = \left\{ z \in \mathbb{C}^{n_0};\ \forall j \in \{1,\dots,n_0\} \quad \widehat{c_5}(a_j) = |a_j|^2 - \frac{1}{n_0} = 0 \right\} \\ \\ \forall x \in \Phi \setminus \Phi_0 \quad c_{i_x}(a,x) \leqslant 0. \end{cases} \tag{II.41}$$

Cela revient à maximiser une fonction de type quadratique, or nous avons vu que ce problème est très difficile.

De plus, le domaine admissible $\widehat{\mathcal{A}}$ défini par la contrainte sur les modules $\widehat{c_5}$ n'est pas non plus convexe.

4.3.3 Idée clé : reformulation sous forme de minimisation convexe

Idée clé : conservation de l'énergie Nous ne pouvons plus reformuler le problème comme en section 3.3.3 page 67, car ici certaines directions en dehors du spot ne sont soumises à aucune contrainte.

Si nous utilisons la même stratégie que dans l'étude préliminaire, en "écrasant" le rayonnement partout hors du spot utile :

- l'absence de contraintes dans la zone hors Terre amènera une partie de l'énergie à se dissiper dans l'espace (voir figure II.22 page 89),
- l'absence de contraintes dans la couronnne autour du spot va empêcher le rayonnement de se concentrer à l'intérieur du spot (voir figure II.23 page 93).

Mais en **rajoutant une contrainte** linéaire notée c_0 **au centre du spot**, nous pouvons à la fois

- concentrer le rayonnement à l'intérieur du spot,
- transformer le problème de maximisation en un problème de minimisation,

En effet, à l'optimum, le rayonnement doit atteindre son maximum au centre du spot noté Φ_0.

Donc, si en plus des contraintes d'isolation, nous imposons à la valeur du champ E en Φ_0 d'être très supérieure au minimum requis dans le spot, en vertu du principe de conservation de l'énergie, comme **nous "écrasons" le rayonnement à l'extérieur du spot là où nous disposons de points de contrôle**, et **nous le "tirons" au centre du spot** vers une valeur assez grande, **nous forçons l'énergie à se concentrer dans le spot**.

Idée clé : contrainte convexe sur les modules Si de plus, **nous écrivons la contrainte des modules sous forme de l'inégalité** $c_5(a_j) := \widehat{c_5}(a_j) \leqslant 0$, **nous définissons un ensemble admissible convexe** $\mathcal{A}$ **pour la variable d'optimisation** a.

Nous espérons obtenir la **saturation de la contrainte sur les modules à l'optimum**, **car les modules vont avoir tendance à être forts pour vérifier le gabarit, c'est-à-dire nous espérons** obtenir l'égalité $c_5(a_j) = 0$ à l'optimum.

Formulation Ainsi, le problème de maximisation $(\mathcal{P}_1)$ se reformule en un problème de minimisation sous contraintes $(\mathcal{P}_2)$:

Le problème de minimisation s'écrit alors

$$(\mathcal{P}_2) \quad \begin{cases} \min\limits_{a\in\mathcal{A}} \max\limits_{x\in\Phi\setminus\Phi_0} \mathcal{D}(a,x) \\ \\ \mathcal{A} = \left\{ z \in \mathbb{C}^{n_0};\ \forall j \in \{1,\dots,n_0\} \quad c_5(a_j) = |a_j|^2 - \frac{1}{n_0} \leqslant 0 \right\} \\ \\ c_0(a) = E(a,x_0) - (g_0 + e_0) = 0 \\ \forall x \in \Phi \setminus \Phi_0 \quad c_{i_x}(a,x) \leqslant 0 \end{cases} \tag{II.42}$$

avec

- x_0 le centre du spot,
- $e_0 \in \mathbb{R}^+$ constante telle que $G_0 := g_0 + e_0$ est supérieur au minimum requis (valeurs avant conversion en dB) par le gabarit dans le spot (voir figure II.23 page 93).

Nous devons **minimiser à présent une fonction de type quadratique convexe sur un domaine de recherche convexe** : nous obtenons un problème de type optimisation convexe plus simple que le problème initial. Il possède une **solution unique** qui correspond au **minimum global du problème initial**.

4.3.4 Reformulation des contraintes

Nous associons des contraintes locales au problème de minimisation (P_2) comme dans la section 3.3.6 page 67.

La contrainte de maximisation à l'intérieur du spot c_1 de $(\mathcal{P}_0)$ est remplacée par la contrainte d'égalité c_0 au centre du spot, et la minimisation du maximum du rayonnement correspond aux contraintes de minimisation hors du spot c_3 et c_4 sous la contrainte de la variable d'optimisation c_5.

Nous obtenons le problème reformulé $(\mathcal{Q}_0)$:

trouver $a \in \mathbb{C}^{n_0}$ tel que

$$(\mathcal{Q}_0) \begin{cases} & c_0(a) & = E(a,x_0) - (g_0+e_0) & = 0 \\ \forall x \in \Phi_2 \setminus \Phi_0 & c_3(a,x) & = \mathcal{D}(a,x) - g_0 + d_1 & \leqslant 0 \\ \forall x \in \Phi_3 \setminus \Phi_2 & c_4(a,x) & = \mathcal{D}(a,x) - g_0 + d_2 & \leqslant 0 \\ \\ \forall j \in 1,\dots,n_0 & c_5(a_j) & = |a_j|^2 - \frac{1}{n_0} & \leqslant 0. \end{cases} \tag{II.43}$$

4.3.5 Mise en oeuvre

Contrôle au bord du spot Comme dans l'étude préliminaire, lors des tests numériques, il est apparu qu'il est nécessaire de contrôler le rayonnement au bord du spot pour atteindre la valeur minimale sur zone utile pour des spots très excentrés.

Mais fixer une valeur constante s'est révélé ne pas être assez efficace, nous imposons plutôt une valeur qui sera mise à jour à chaque itération.

Nous rajoutons donc la contrainte d'égalité suivante qui ne change pas la nature du problème (même type de contrainte que c_0) :

$$\forall x \in \partial\Phi_0 \quad E(a,x) = G_m \tag{II.44}$$

avec

- $\partial\Phi_0$ les points du bord du spot Φ_0 (voir figure II.23),
- G_m est la moyenne des valeurs $\{E(a,x),\ x \in \partial\Phi_0\}$ à l'itération précédente.

Ainsi la valeur G_m imposée au bord du spot $\partial\Phi_0$ augmente au fur et à mesure que le rayonnement est dépointé dans le spot, et tire vers le haut les valeurs du rayonnement sur $\partial\Phi_0$. Cela permet de contrôler correctement le rayonnement au cours des itérations malgré l'absence de points de maillage autour du spot (voir figure II.23).

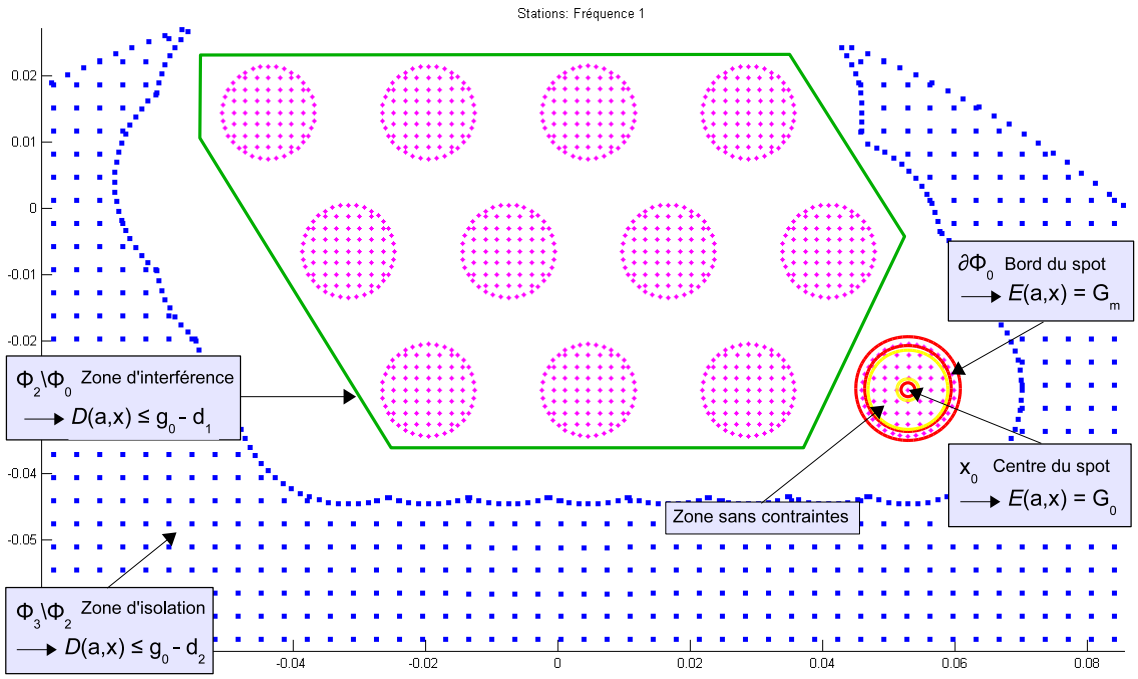


FIG. II.23 – Domaines pour les contraintes modifiées

Il est en fait délicat d'imposer la même valeur du champ E égale à G_m sur tout le bord du spot, mais dans notre cas les résultats numériques sont corrects : grâce à la faible taille du spot et à la symétrie par rapport à son centre, la loi est localement symétrique.

Pour des spots plus larges ou de forme irrégulière, il faudra améliorer cette contrainte sur le bord (voir la proposition (III.21) dans la conclusion).

Contraintes finales Pour une antenne à géométrie donnée (n ER) et pour un spot k fixé, le problème reformulé final à résoudre avec les contraintes renommées C_j est :

trouver $a \in \mathbb{C}^{n_0}$ tel que

$$(\mathcal{Q}_1)\begin{cases} & C_1(a) & = E(a,x_0) - (g_0 + e_0) & = 0 \\ \forall x \in \partial\Phi_0 & C_2(a,x) & = E(a,x) - G_m & = 0 \\ \forall x \in \Phi_2 \setminus \Phi_0 & C_3(a,x) & = \mathcal{D}(a,x) - g_0 + d_1 & \leqslant 0 \\ \forall x \in \Phi_3 \setminus \Phi_2 & C_4(a,x) & = \mathcal{D}(a,x) - g_0 + d_2 & \leqslant 0 \\ \\ \forall j \in 1,\ldots,n_0 & C_5(a_j) & = |a_j|^2 - \frac{1}{n_0} & \leqslant 0. \end{cases} \tag{II.45}$$

4.3.6 Fonction coût

Nous procédons de manière analogue à l'étude préliminaire en section 3.3.8 page 69.

Evaluation des contraintes par discrétisation Nous définissons une fonction $\widetilde{F} : \mathbb{R}^{2n_0} \longrightarrow \mathbb{R}^{q+n}$ qui évalue les contraintes. Nous discrétisons le problème en x en utilisant le maillage donné.

Par rapport à l'étude précédente, nous intégrons en plus la contrainte sur les modules par pénalisation de chacune de ses composantes sur les n ER.

Nous obtenons :

$$\widetilde{F}(a) = \begin{pmatrix} \widetilde{F}_1(a) \\ \vdots \\ \widetilde{F}_{p+n}(a) \end{pmatrix} = \begin{pmatrix} \widehat{C}_{i_1}(a,x_1) \\ \vdots \\ \widehat{C}_{i_p}(a,x_p) \\ \ldots\ldots \\ C_5^+(a_1) \\ \vdots \\ C_5^+(a_n) \end{pmatrix} \tag{II.46}$$

$$\widehat{C}_{i_j}(a,x_j) = C_{i_j}(a,x_j) \text{ ou } C_{i_j}(a,x_j)^+$$

avec

- Les composantes correspondent aux p points du maillage Terre sur lesquels on a défini des contraintes, et aux n valeurs de module de l'alimentation a.
- Chaque point x_j appartient à un des quatre domaines disjoints associés aux contraintes C_i :

  on note $i_j \in \{1,\ldots,4\}$ l'indice correspondant et C_{i_j} est la contrainte qu'on veut vérifier au point x_j.
- En ce qui concerne les contraintes de gabarit :
  - si $i_j = 1$ ou 2, c'est-à-dire x_j est au centre ou sur le bord du spot utile, on doit vérifier en x_j une contrainte d'égalité (C_1 ou C_2), alors $\widehat{C}_{i_j}(a,x_j) = C_{i_j}(a,x_j)$,

- si $i_j = 3$ ou 4, on doit vérifier en x_j une contrainte d'inégalité (C_3 ou C_4), alors $\widehat{C}_{i_j}(a, x_j) = C_{i_j}(a, x_j)^+$.

Ainsi dans tous les cas, pour chaque composante de $\widetilde{F}$:

- $\widetilde{F}_j > 0$ quand la contrainte C_{i_j} correspondante n'est pas vérifiée,
- $\widetilde{F}_j = 0$ quand la contrainte C_{i_j} correspondante est vérifiée.

Pondération Nous construisons la fonction F en pondérant les composantes de $\widetilde{F}$:

$$F(a) = \begin{pmatrix} F_1(a) \\ \vdots \\ F_{p+n}(a) \end{pmatrix} = \begin{pmatrix} \lambda_{i_1}\ \widetilde{F}_1(a) \\ \vdots \\ \lambda_{i_p}\ \widetilde{F}_p(a) \\ \ldots\ldots \\ \lambda_5\ \widetilde{F}_{p+1}(a) \\ \vdots \\ \lambda_5\ \widetilde{F}_{p+n}(a) \end{pmatrix} \tag{II.47}$$

avec

- cinq poids $\lambda_j \in \mathbb{R}^+$, chacun associé à une des contraintes $C_j,\ j \in \{1, \ldots, 5\}$.

Fonction à minimiser On définit la fonction coût à minimiser $J : \mathbb{R}^{2\ n_0} \longrightarrow \mathbb{R}^+$ en prenant la norme au carré de F :

$$J(a) = \frac{1}{2} \left(\sum_{j=1}^{p} \left[\widehat{C}_{i_j}(a, x_j) \right]^2 + \sum_{j=1}^{n_0} \left[C_5(a_j)^+ \right]^2 \right). \tag{II.48}$$

Minimiser J revient à chercher la loi d'alimentation solution du problème transformé $(\mathcal{Q}_1)$. Le problème initial et le problème reformulé ne sont pas strictement équivalents du point de vue mathématique, mais leurs solutions numériques sont numériquement très proches. Il faut ainsi vérifier *a posteriori* que la solution trouvée est aussi solution du problème $(\mathcal{P}_0)$.

4.3.7 Algorithme

Nous conservons le même algorithme d'optimisation que dans l'étude préliminaire (voir section 3.3.9 page 71).

Alimentation optimale : cas non relaxé A la fin de l'algorithme d'optimisation, nous obtenons l'alimentation optimale que nous désignons par le "cas non relaxé", où la contrainte sur les modules doit être respectée.

Alimentation optimale : cas non relaxé projeté Nous rappelons que les variables d'optimisation sont les parties réelles et imaginaires, on ne peut donc pas avec cette approche fixer les modules à leur valeur cible et optimiser uniquement par rapport aux phases.

Concernant l'alimentation optimale trouvée, les contraintes d'égalité sur les modules $|a_j|$ ne sont peut être pas toutes respectées, ou avec une certaine marge. En effet, nous utilisons une méthode d'optimisation sans contraintes, les inégalités (ou contraintes de gabarit) et la contrainte sur les modules du problème de synthèse de réseau sont intégrées par pénalisation à la fonction coût. L'algorithme travaille alors dans un espace de recherche plus large que le domaine admissible, et la solution trouvée ne respecte strictement les contraintes de gabarit et de module que si la fonction coût vaut 0 (cas difficile à obtenir en pratique).

Or la contrainte sur les modules a pour origine une contrainte physique sur l'alimentation qu'on veut appliquer à l'antenne lors de l'utilisation : nous devons fournir un résultat qui la respecte exactement.

Pour ces raisons, à la fin de l'algorithme, quand nous convertissons le vecteur des parties réelles et imaginaires en vecteur des modules et phases :

- nous projetons les modules de l'alimentation obtenue sur les valeurs cibles,
- et nous gardons les phases optimales.

Nous dirons de cette alimentation qu'elle correspond au "cas non relaxé projeté".

Alimentation optimale : cas relaxé Nous allons appliquer l'algorithme lors des tests en ignorant la contrainte sur les modules afin de disposer d'un degré de liberté supplémentaire dans l'espace de recherche. Dans ce cas nous désignons l'alimentation obtenue par le "cas relaxé".

Ces alimentations serviront à définir un critère pour construire la fonction coût du problème d'optimisation topologique.

Remarque Le cas relaxé correspond à la modélisation d'une antenne pour laquelle le dispositif électronique branché sur chaque ER permet de contrôler à la fois le module et la phase d'alimentation.

Le cas non relaxé correspond à la modélisation d'une antenne pour laquelle le dispositif électronique branché sur chaque ER permet de contrôler uniquement la phase d'alimentation : cette approche, moins coûteuse au niveau de la fabrication, est celle souhaitée par Thalès pour concevoir l'antenne.

4.3.8 Affichages

Les affichages sont les mêmes que dans la section 3.3.10 page 72.

Niveaux de directivité Nous donnons une précision sur l'échelle de couleurs qui permet ici de séparer les trois valeurs de gabarit. Le gabarit est respecté si à l'affichage :

- tous les points de la zone utile sont dans une zone rouge (couleur en haut de l'échelle),
- aucun point de la zone d'interférence n'est dans une zone verte (couleur au milieu de l'échelle),
- aucun point de la zone d'isolation n'est dans une zone bleue (couleur en bas de l'échelle).

4.4 Résultats numériques

4.4.1 Cas non relaxé

Nous effectuons la recherche des lois d'alimentations optimales pour les 44 spots de la zone de couverture.

Les paramètres d'entrée de l'algorithme sont :

- poids fixes sur les contraintes : $\lambda_j = 10,\ j = 1 \dots 4$ pour les contraintes de gabarit et $\lambda_5 = 10^4$ pour les contraintes de module,
- nombre maximum d'itérations : 20,
- alimentation initiale a_0 vecteur constant.

Performances La figure II.24 présente les résultats dans le cas non relaxé pour l'ensemble des spots, avant et après projection sur les valeurs cibles.

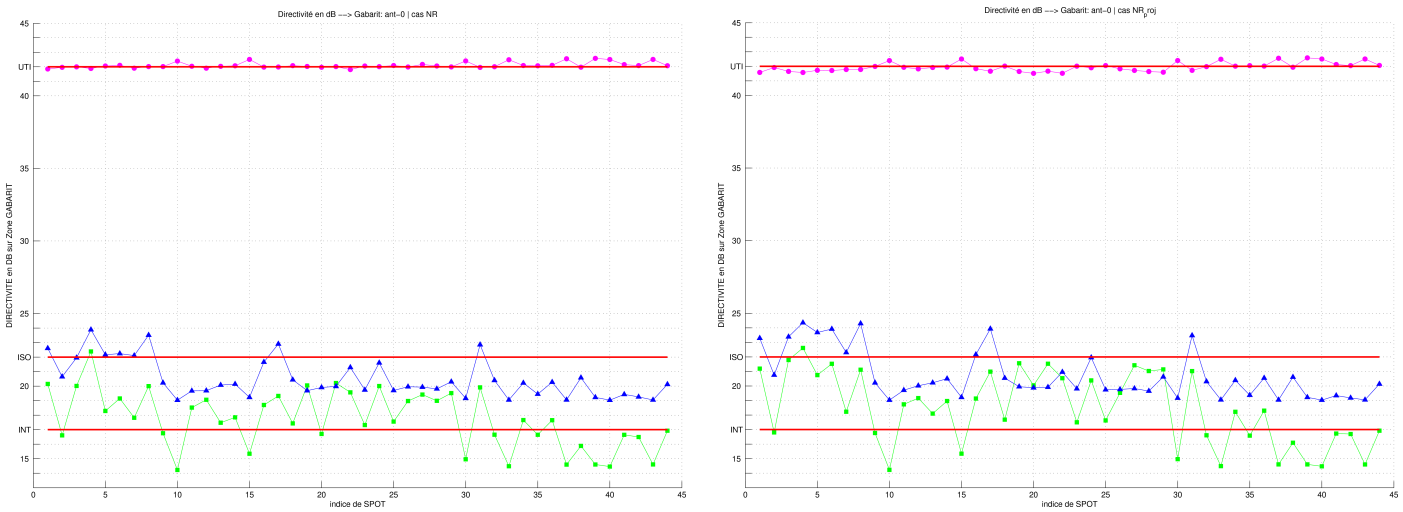


FIG. II.24 – Performances pour l'antenne de départ Ant_0
A gauche dans le cas non relaxé
A droite dans le cas non relaxé projeté

Les alimentations trouvées ne respectent pas les contraintes de gabarit sur tous les spots, et le résultat est dégradé après projection des modules sur les valeurs cibles.

Analyse Prenons par exemple l'alimentation optimale du spot n°1, et nous affichons pour l'alimentation optimale dans le cas non relaxé :

- les niveaux de directivité dans la figure II.26,
- les cartes de module et de phase dans la figure II.25.

Nous constatons que la puissance n'est pas concentrée dans le spot : des remontées du rayonnement apparaissent dans la zone proche autour du spot.

La carte des modules est uniforme puisque ce sont les valeurs cibles, mais on voit que quelques ER au centre ont une valeur trop importante. La carte des phases pésente des bandes parallèlles moins nettes que dans l'étude préliminaire.

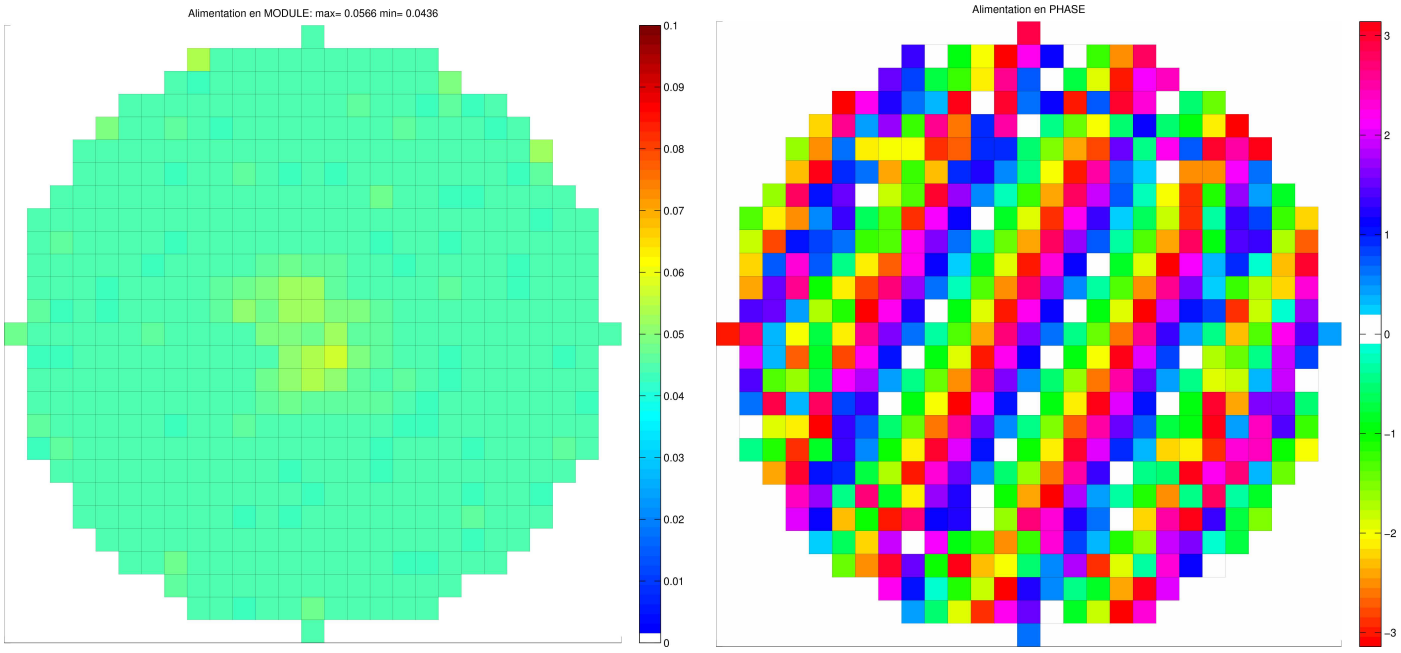


FIG. II.25 – Spot n°1 : alimentation optimale pour Ant_0 dans le cas non relaxé
A gauche : carte des modules
A droite : carte des phases

Toujours pour le spot n°1, nous visualisons en fonction des itérations dans la figure II.27 :

- la valeur de la fonction coût normalisée $\frac{J_k}{J_0}$,
- les niveaux de directivité D sur les zones de gabarit.

Nous observons que le gabarit est respecté après dix itérations, mais ensuite la valeur de la directivité remonte sur la zone d'interférence pour dépasser la valeur du gabarit (22 dB valeur maximum), pourtant la fonction coût diminue tout au long des itérations. Cela n'est pas contradictoire, car la composante de F correspondant à la contrainte sur les modules n'est pas affichée ici. Or l'algorithme fait d'abord tendre vers 0 les erreurs liées aux trois zones de gabarit, puis ensuite il diminue l'erreur liée aux modules, mais il le fait au détriment d'une des erreurs précédentes (sur la zone d'interférence), si bien que globalement J continue de diminuer.

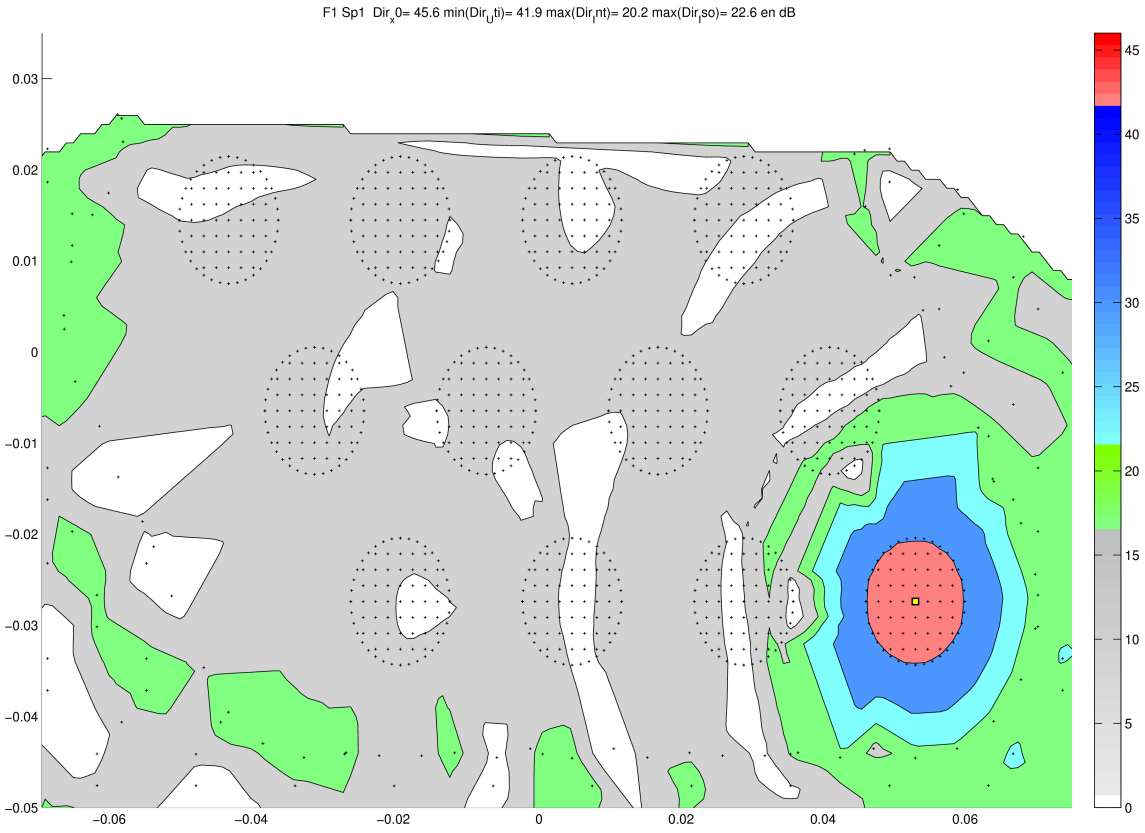


FIG. II.26 – Spot n°1 : alimentation optimale pour Ant_0 dans le cas non relaxé
Courbes iso-niveaux de la directivité

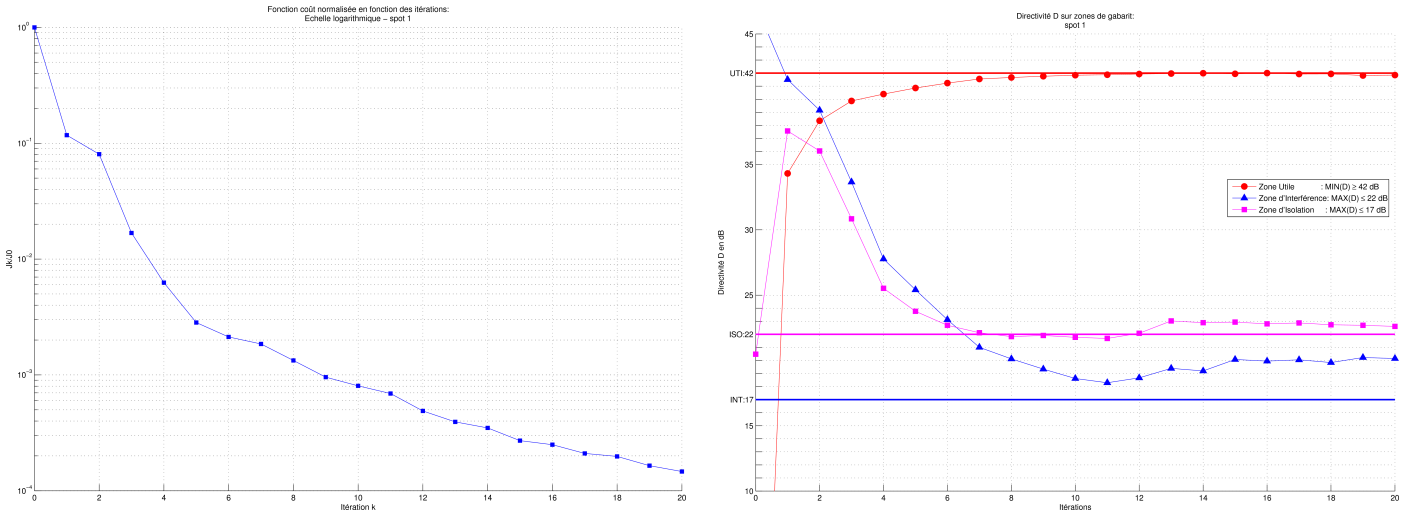


FIG. II.27 – Spot n°1 pour Ant_0 dans le cas non relaxé
A gauche : fonction coût normalisée $\frac{J_k}{J_0}$ en fonction des itérations
A droite : directivité D (dB) sur zones de gabarit en fonction des itérations

Pour le spot n°1, nous affichons les cartes de module et de phase dans la figure II.28 pour l'alimentation dans le cas non relaxé au bout de 10 itérations (contre 20 pour l'alimentation optimale).

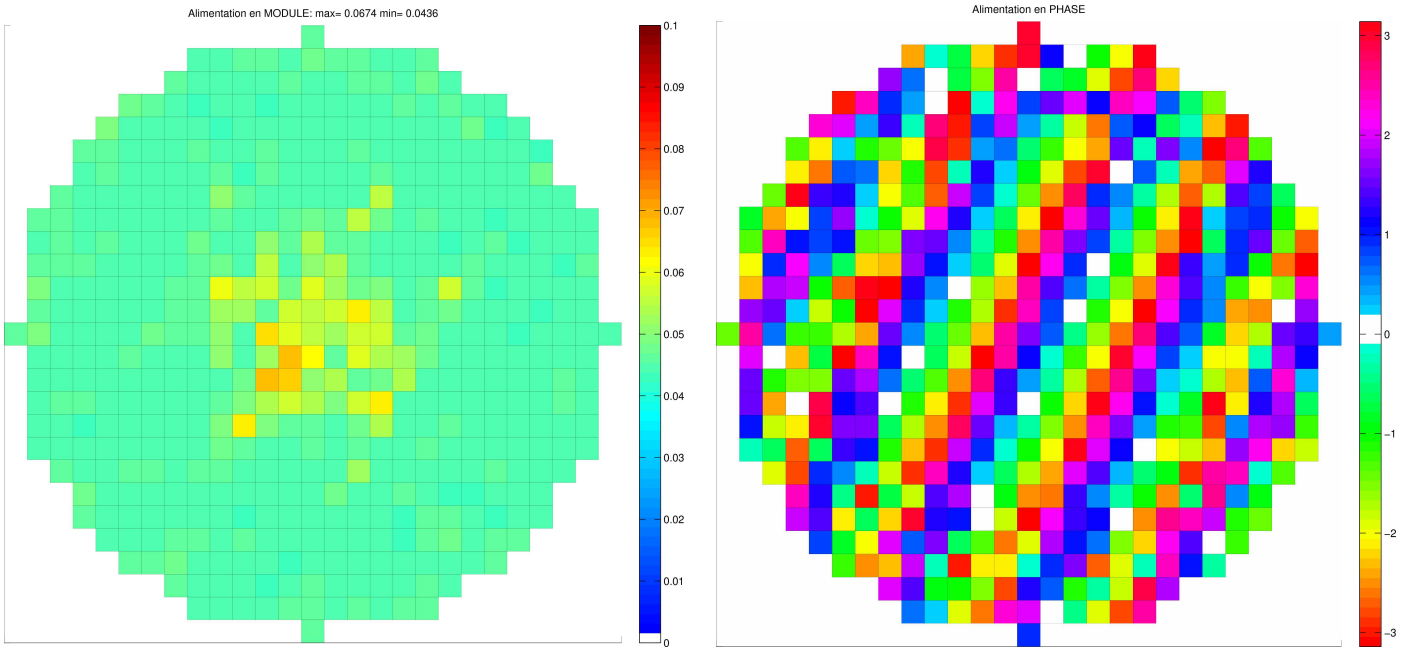


FIG. II.28 – Spot n°1 : alimentation au bout de 10 itérations pour Ant_0 dans le cas non relaxé
A gauche : carte des modules
A droite : carte des phases

Nous observons beaucoup plus de valeurs de module au-dessus de la valeur cible que pour l'alimentation après 20 itérations.

4.4.2 Cas relaxé

La contrainte sur les modules ne permet pas d'atteindre les performances demandées en jouant uniquement sur les phases, et nous avons vu que vérifier la contrainte sur les modules dégrade les performances. Nous allons relâcher cette contrainte et voir comment se comporte l'algorithme dans ce cas.

Les paramètres d'entrée de l'algorithme sont :

- poids fixes sur les contraintes : $\lambda_j = 10,\ j = 1 \dots 4$ pour les contraintes de gabarit et $\lambda_5 = 0$ pour la relaxation des contraintes de module,
- nombre maximum d'itérations : 20,
- alimentation initiale a_0 vecteur constant.

Performances La figure II.29 présente les résultats dans le cas relaxé pour l'ensemble des spots.

Analyse Cette fois, une dizaine d'itérations sont suffisantes pour trouver les alimentations optimales respectant le gabarit dans tous les cas.

Nous comparons les résultats avec le cas précédent sur le spot n°1 en affichant pour l'alimentation optimale dans le cas relaxé

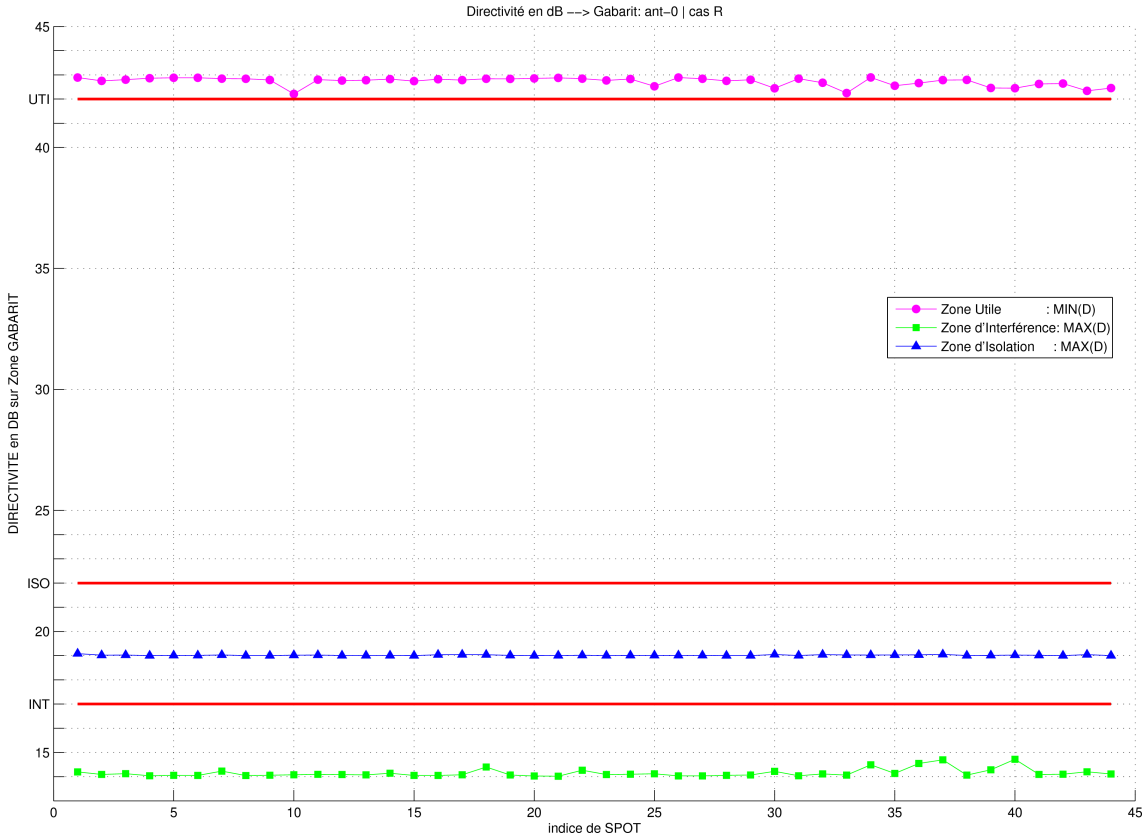


FIG. II.29 – Performances pour Ant_0 dans le cas relaxé

- dans la figure II.31 les niveaux de rayonnement,
- dans la figure II.30 les cartes de modules et phases.

Le degré de liberté supplémentaire est utilisé par l'algorithme qui place des valeurs de module plus fortes au centre de l'antenne et sur quelques ERel du bord , et des valeurs très faibles sur une couronne d'ERel ; de plus les phases pésentent des bandes parallèles beaucoup plus régulières.

La puissance est beaucoup plus concentrée à l'intérieur du spot que dans le cas non relaxé : le maximum est ici de 47dB contre 45 dB. En valeurs avant la conversion en décibels (utilisées par l'algorithme), cela représente un maximum de l'ordre de 5.10^4 contre 3.10^4 et la valeur minimum à atteindre est $1,5.10^4$.

Nous présentons à présent en fonction des itérations dans la figure II.32

- la valeur de la fonction coût normalisée $\frac{J_k}{J_0}$,
- les niveaux de directivité D sur les zones de gabarit,

Les alimentations optimales trouvées dans le cas des contraintes de module relaxées respectent le gabarit avec de la marge, contrairement au cas non relaxé.

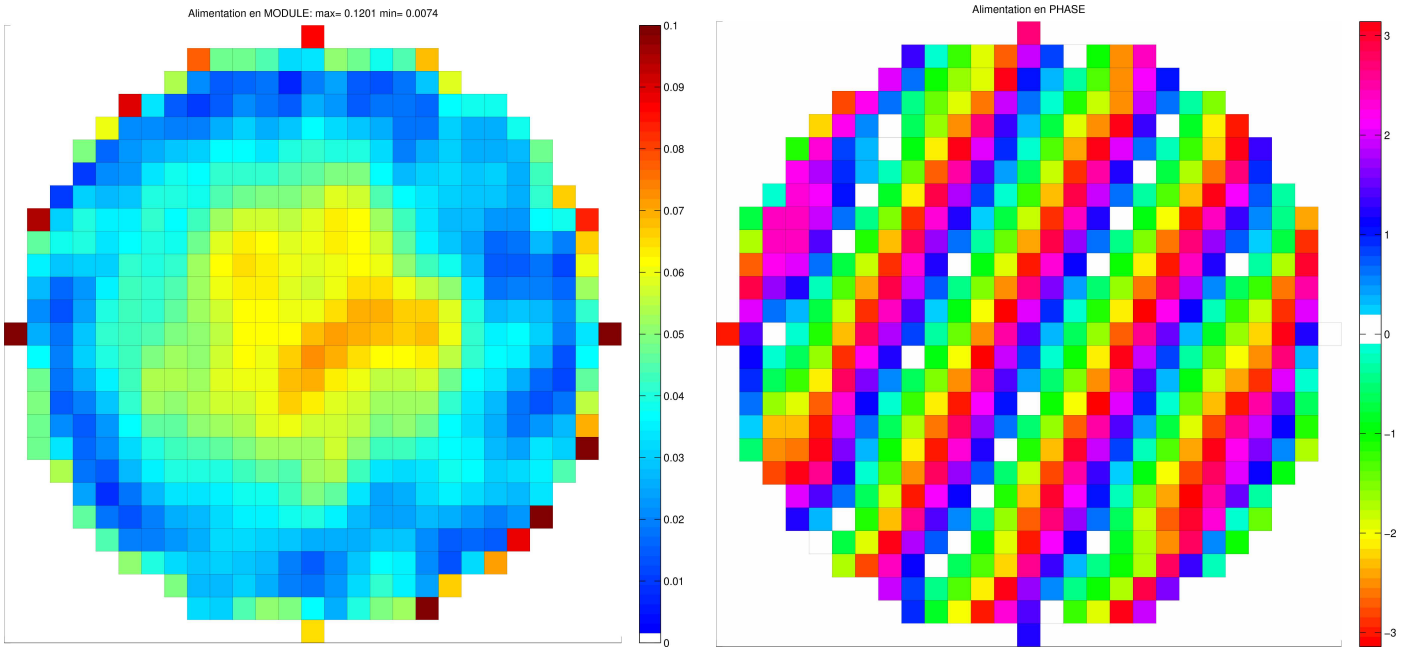


FIG. II.30 – Spot n°1 : alimentation optimale pour Ant_0 dans le cas relaxé
A gauche : carte des modules
A droite : carte des phases

Conclusion Nous allons exploiter ce résultat dans la partie où nous allons chercher à grouper les ERel tout en essayant de conserver les performances : il faudra extraire l'information donnée par les modules relaxés et l'utiliser pour décider les regroupements.

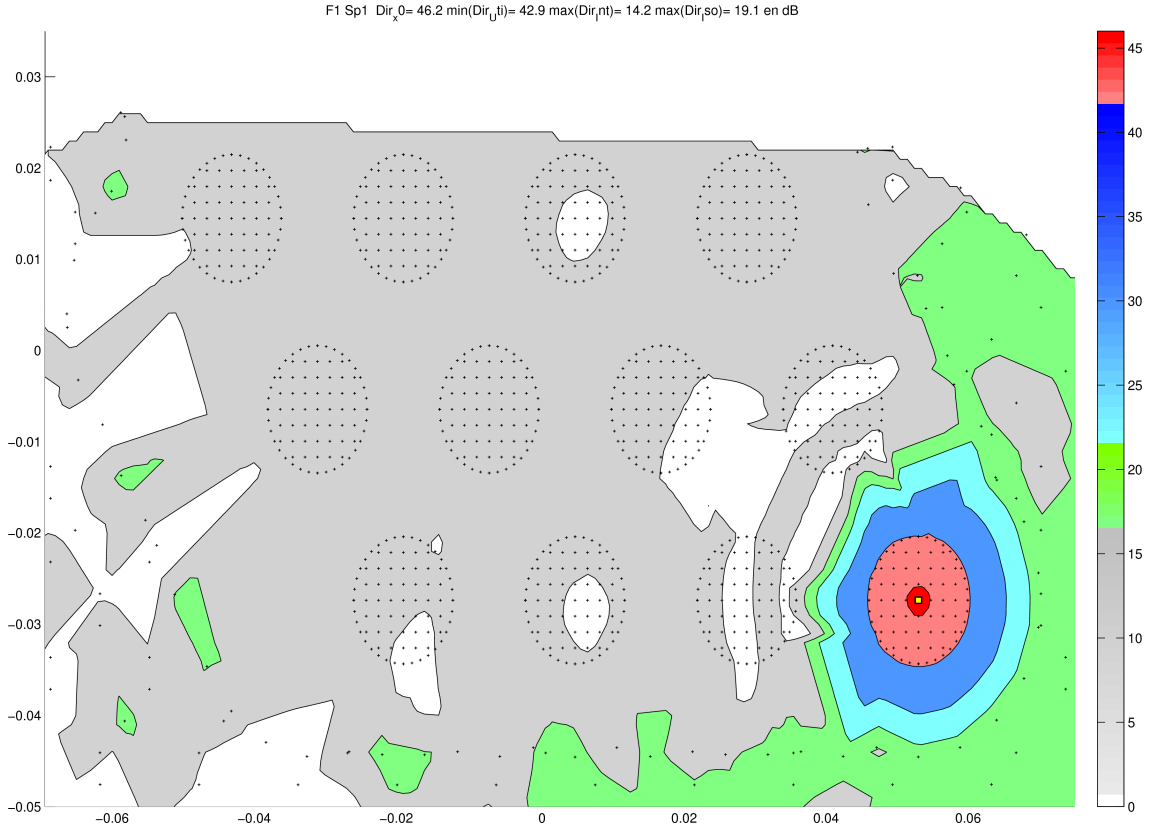


FIG. II.31 – Spot n°1 : : alimentation optimale pour Ant_0 dans le cas rrelaxé
Courbes iso-niveaux de la directivité pour l'alimentation optimale

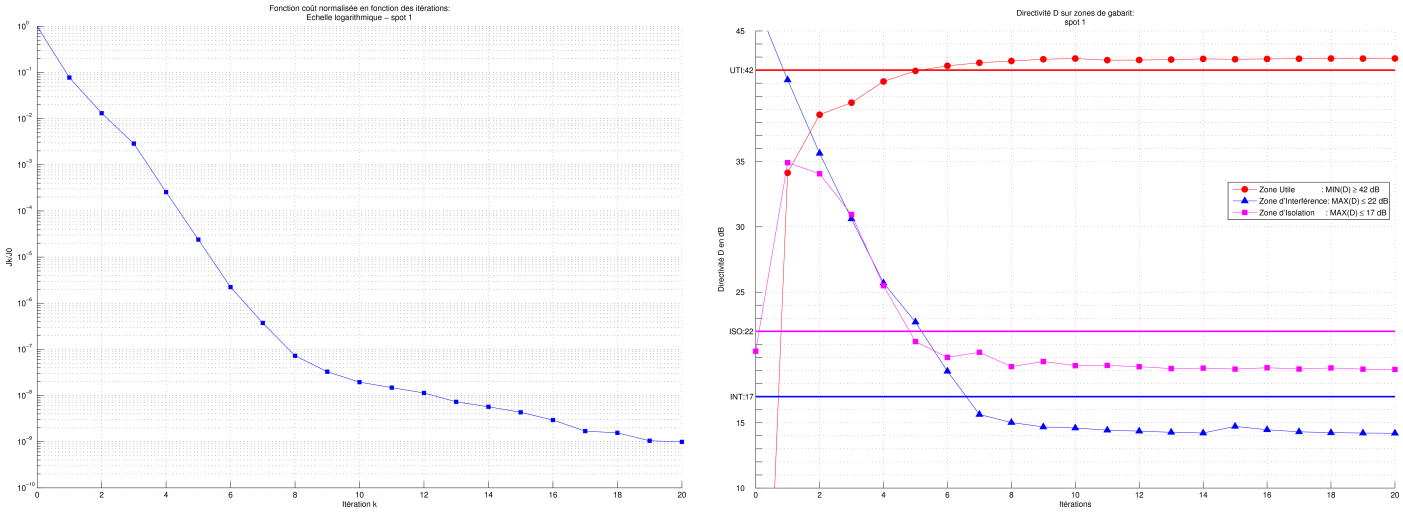


FIG. II.32 – Spot n°1
A gauche : fonction coût normalisée $\frac{J_k}{J_0}$ en fonction des itérations
A droite : directivité D (dB) sur zones de gabarit en fonction des itérations

Chapitre III

Optimisation topologique de l'antenne

Sommaire

1 Introduction

Le problème consiste à chercher une nouvelle géométrie pour l'antenne qui doit :

- présenter un nombre de sources (ou ER) le plus inférieur possible à celui de l'antenne de départ,
- respecter les contraintes de gabarit et sur les modules définies dans le problème du chapitre précédent en section 4.2.3 page 88.

Il n'existe pas de méthode générale pour résoudre ce problème (voir [CCG+07]) :

plusieurs approches sont envisageables (voir section 2.3.2 page 25), mais dans tous les cas, les algorithmes envisagés ne permettent pas de définir une géométrie valable pour l'ensemble des spots utiles.

Le parti pris est ici le suivant : l'antenne de départ a une maille fine et régulière avec de nombreux ERel, permettant de satisfaire au mieux le gabarit, et la nouvelle géométrie de l'antenne finale sera obtenue par regroupement de ces ERel.

Le maillage des ER qui est régulier au départ deviendra irrégulier pour l'antenne optimisée : casser la régularité du réseau permettra d'abaisser les lobes de réseau.

L'idée majeure est de calculer d'abord les lois optimales pour tous les spots utiles avec l'antenne initiale dans le cas des contraintes sur les modules relaxées. Nous obtenons ensuite un résumé de ces modules optimaux en appliquant une technique de réduction de données au résultat (décomposition en valeurs singulières).

Nous construisons la nouvelle géométrie lors d'un processus itératif.

Pour les antennes avec regroupements, nous simulons une alimentation de calcul où les contrôles sont répartis sur l'ensemble des éléments de départ sous-jacents (ERel) : le critère à optimiser est l'erreur entre les valeurs de contrainte sur les modules de l'antenne regroupée, qui dépend de la géométrie, et les modules optimaux obtenus sur les ERel.

C'est une méthode de type gradient topologique : on regroupe les ERel aux endroits où l'erreur est la plus forte pour faire diminuer le critère.

Ainsi, le degré de liberté perdu lors de l'application de l'alimentation finale respectant la contrainte sur les modules (cas non relaxé) est alors compensé par le fait que les valeurs à imposer sont proches des valeurs souhaitées pour les meilleures performances possibles.

Nous présentons alors l'antenne optimisée obtenue après l'application de cette méthode puis la validation de ses performances (voir aussi [TA08]).

Dans la suite nous considérons les hypothèses de l'étude industrielle (voir section 2.1 du chapitre précédent page 52), concernant le calcul de la directivité et les contrainte d'alimentation.

2 Regroupement d'ERel

2.1 Panel de motifs d'ER

Les ER peuvent être fabriqués industriellement avec des coûts de fabrication modérés pour certain nombre de motifs standards. Nous avons alors décidé de nous limiter à priori à un panel de motifs défini en concertation avec Thalès, c'est-à-dire de les intégrer comme des contraintes de l'étude.

Le panel est le suivant (voir figure III.1) :

- des ER carrés et rectangles constitués de 2,3 ou 4 ERel,
- des ER en forme de coudes constitués de 3 ERel.

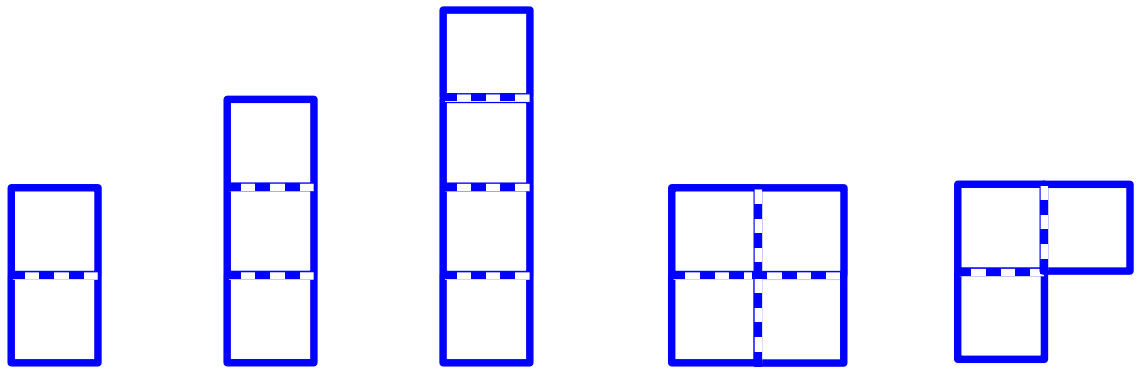


FIG. III.1 – Panel des motifs d'ER réalisables

2.2 Antenne regroupée

2.2.1 Alimentation sur les ERel

Soit l'antenne de départ Ant_0 constituée d'un réseau régulier de n_0 ERel. On regroupe certains ERel voisins pour former une nouvelle antenne Ant_k constituée d'un nombre $n \leqslant n_0$ d'ER.

On définit pour Ant_k :

- le vecteur alimentation $a \in \mathbb{C}^n$ avec pour un indice d'ER $j \in \{1, \ldots, n\}$ $a_j = A_j \, e^{i\varphi_j}$,
- le vecteur alimentation sur les ERel $\widehat{a} \in \mathbb{C}^{n_0}$ avec la règle suivante :
  soit $i \in \{1, \ldots, n_0\}$ l'indice d'un ERel sous-jacent dans Ant_k, et soit $j \in \{1, \ldots, n\}$ l'indice correspondant de l'ER (regroupé ou pas) auquel il appartient, alors le module et la phase sont égaux.
  C'est à dire si $a_j = A_j \, e^{i\varphi_j}$ alors $\widehat{a}_i = A_j \, e^{i\varphi_j}$

Cette propriété est illustrée par un exemple dans la figure III.2 avec des regroupements d'ERel par 2 et par 4.

2.2.2 Calcul de la directivité

Nous pouvons définir deux vecteurs champ :

- $\widehat{E}(x_k) = (E_1(x_k), \ldots, E_{n_0}(x_k))^T \in \mathbb{C}^{n_0}$ les valeurs de champ pour les n_0 ERel de départ au point x_k fournies par le fichier de données,
- $E(x_k) = (E_1(x_k), \ldots, E_n(x_k))^T \in \mathbb{C}^n$ le vecteur champ obtenu à partir des valeurs du vecteur champ précédent avec la règle suivante :
  soit $j \in \{1, \ldots, n\}$ un indice d'ER dans Ant_k, composé d'un regroupement de r_j ERel, $1 \leqslant r_j \leqslant 4$, et $\{i_1, \ldots, i_{r_j}\} \in \{1, \ldots, n_0\}$ les indice des ERel correspondants,
  alors $E_j(x_k) = \sum_{i=i_1}^{i_{r_j}} \widehat{E}_i(x_k)$.

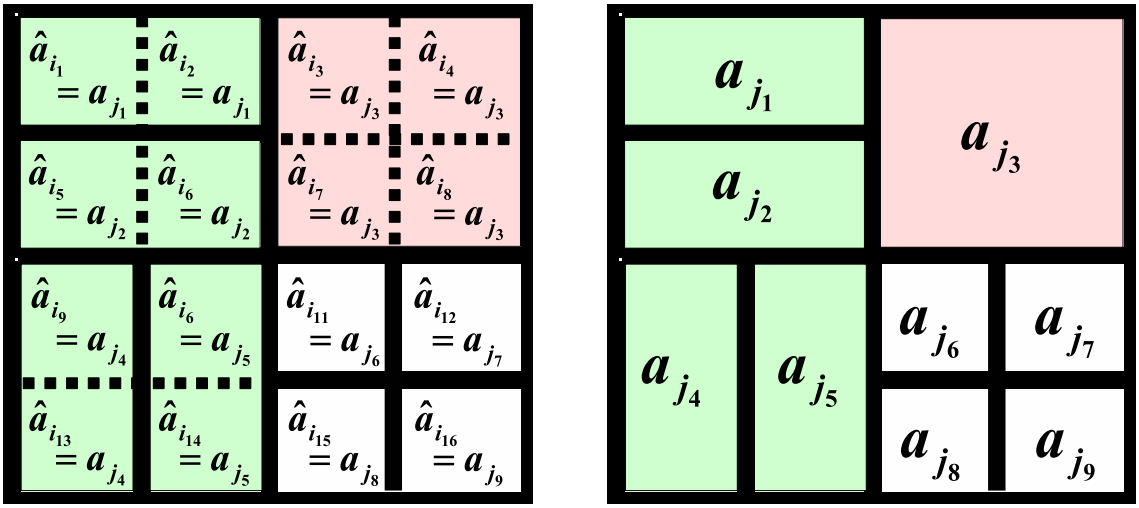


FIG. III.2 – Alimentation d'une antenne avec regroupements Ant_k à n ER
A gauche : alimentation $\widehat{a} \in \mathbb{C}^{n_0}$ sur les ERel sous-jacents
A droite : alimentation $a \in \mathbb{C}^n$ sur les ER

En effet, ainsi nous obtenons le champ rayonné total en x_k pour une antenne avec regroupements de deux manières :

- en utilisant l'alimentation sur les ERel $\widehat{a} \in \mathbb{C}^{n_0}$ avec $\widehat{E}(x_k)$,
- ou bien en utilisant l'alimentation sur les ER $a \in \mathbb{C}^n$ avec $E(x_k)$.

Nous obtenons alors d'après la définition de $\widehat{a}$ vue précédemment :

$$\begin{aligned} E(a,x_k) &= \sum_{j=1}^{n} a_j \; E_j(x_k) \\ &= \sum_{j=1}^{n} a_j \left(\sum_{i=i_1}^{i_{r_j}} \widehat{E}_i(x_k) \right) \\ &= \sum_{j=1}^{n} \left(\sum_{i=i_1}^{i_{r_j}} a_j \widehat{E}_i(x_k) \right) \\ &= \sum_{j=1}^{n} \left(\sum_{i=i_1}^{i_{r_j}} \widehat{a}_i \widehat{E}_i(x_k) \right) \\ &= \sum_{i=1}^{n_0} \widehat{a}_i \; \widehat{E}_i(x_k). \end{aligned} \tag{III.1}$$

Nous rappelons que la directivité en un point x_k pour l'alimentation $a \in \mathbb{C}^n$ se calcule comme le module du champ E au carré (voir section 2.1 page 52 du chapitre II) :

$$\mathcal{D}(a,x_k) = |E(a,x_k)|^2 = \left| \sum_{j=1}^{n} a_j E_j(x_k) \right|^2 . \tag{III.2}$$

2.2.3 Calcul des lois optimales

Avec la formule précédente de calcul de la directivité, nous pouvons alors appliquer directement l'algorithme du chapitre précédent afin de calculer les lois d'alimentations optimales $a \in \mathbb{C}^n$ de Ant_k.

2.2.4 Contrainte sur les modules : valeurs cibles

Nous rappelons que les valeurs cibles sont les valeurs de contrainte pour les modules, ce qui pour l'alimentation $a \in \mathbb{C}^{n_0}$ de Ant_0 se traduit par

$$\forall i = 1, \ldots, n_0 \quad |a_i| = \frac{1}{\sqrt{n_0}}. \tag{III.3}$$

Cette contrainte est issue de la contrainte physique selon laquelle la puissance totale d'alimentation est égale à un.

Cela s'écrit ici pour Ant_k et son alimentation sur les ERel $\widehat{a} \in \mathbb{C}^{n_0}$

$$\sum_{i=1}^{n_0} |\widehat{a_i}|^2 = 1. \tag{III.4}$$

Dans Ant_k, un seul dispositif électronique de contrôle est relié à chaque ER regroupé, et la puissance d'alimentation est répartie uniformément sur chacun ; on obtient la contrainte suivante :

$$\forall i = 1, \ldots, n_0 \quad |\widehat{a_i}| = \frac{1}{\sqrt{r_i}\sqrt{n}} \tag{III.5}$$

avec comme précédemment : $1 \leqslant r_i \leqslant 4$ est le nombre d'ERel qui forment l'ER regroupé auquel appartient l'ERel d'indice $i \in \{1 \ldots n_0\}$.

En effet, ainsi la relation de puissance totale (III.4) est vérifiée avec $\widehat{a}$ car le module et la phase sont égaux sur un ER et sur les ERel sous-jacents :

$$\begin{aligned}
\sum_{i=1}^{n_0} |\widehat{a_i}|^2 &= \sum_{i=1}^{n_0} \left(\frac{1}{\sqrt{r_i}\sqrt{n}}\right)^2 \\
&= \tfrac{1}{n} \sum_{i=1}^{n_0} \frac{1}{r_i} \\
&= \tfrac{1}{n} \sum_{j=1}^{n} \left(\sum_{i=i_1}^{i_{r_j}} \frac{1}{r_i} \right) \\
&= \tfrac{1}{n} \sum_{j=1}^{n} 1 \\
&= \tfrac{1}{n}\, n \\
&= 1.
\end{aligned} \tag{III.6}$$

Nous obtenons finalement les valeurs cibles à respecter pour l'alimentation $a \in \mathbb{C}^n$ de Ant_k

$$\forall j = 1, \ldots, n \quad |a_j| = \frac{1}{\sqrt{r_j}\sqrt{n}}. \tag{III.7}$$

Nous vérifions à nouveau la relation (III.3) avec a cette fois, sachant que la puissance sur un ER regroupé est proportionnelle aux nombre d'ERel le composant :

$$\begin{aligned} \sum_{j=1}^{n} |a_j|^2 &= \sum_{j=1}^{n} r_j \left| \frac{1}{\sqrt{r_j}\sqrt{n}} \right|^2 \\ &= \sum_{j=1}^{n} \frac{1}{n} \\ &= 1. \end{aligned} \tag{III.8}$$

Nous proposons un exemple avec un détail de l'antenne de départ Ant_0 puis d'une antenne Ant_k obtenue avec certains regroupements (par 2 ou par 4), les valeurs cibles correspondantes dans les figures III.3 et III.4 .

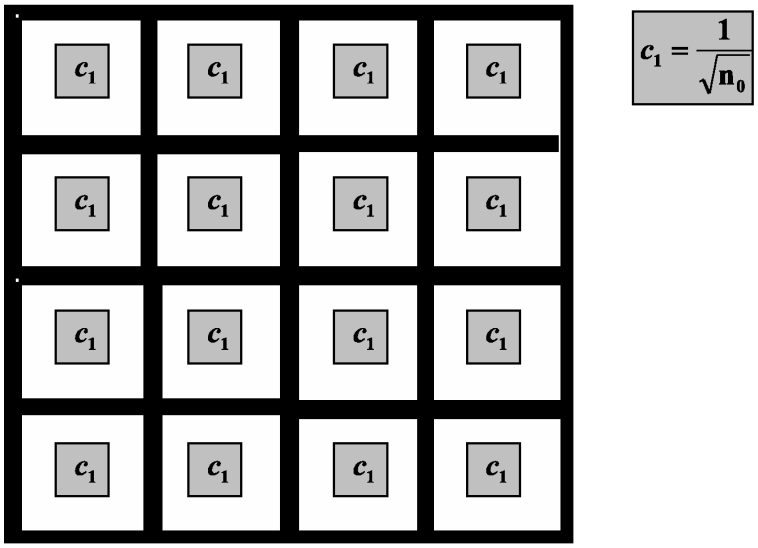


FIG. III.3 – Antenne Ant_0, n_0 ERel : valeurs cibles

Nous utiliserons dans la suite les valeurs cibles :

- pour a dans le calcul des lois optimales (contrainte à respecter sur les modules, voir section 4.3.6 page 94 du chapitre précédent),
- pour $\widehat{a}$ dans le calcul de la fonction coût de l'optimisation topologique (voir section 3.3 page 115).

3 Méthode

3.1 Idée clé : relaxation de la contrainte sur les modules

On suppose que le nombre d'ER de Ant_k est n avec l'hypothèse

$$\frac{n_0}{2} \leqslant n \leqslant n_0. \tag{III.9}$$

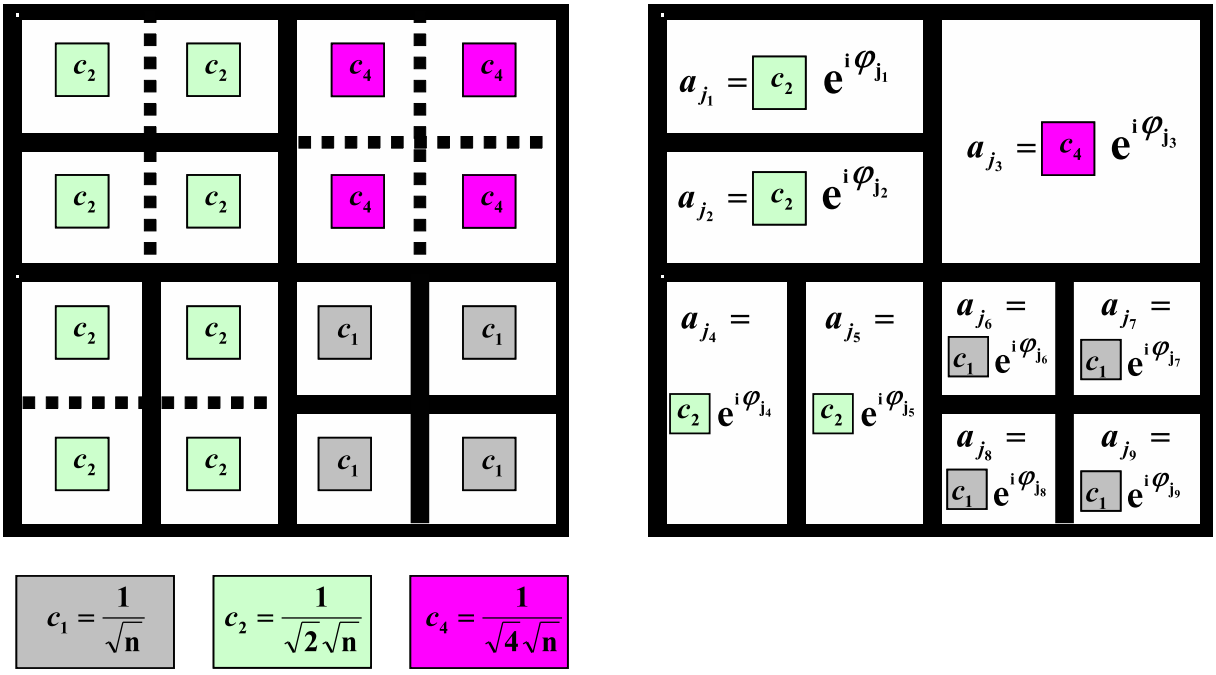


FIG. III.4 – Antenne Ant_k, n ER
A gauche : valeurs cibles sur les ERel
A droite : alimentation respectant les valeurs cibles (cas non relaxé)

On en déduit

$$\frac{1}{2}\frac{1}{\sqrt{n}} \leqslant \frac{1}{\sqrt{n_0}} \leqslant \frac{1}{\sqrt{n}}. \tag{III.10}$$

D'après la section précédente, les valeurs cibles de l'alimentation sur les ERel de Ant_k auront les propriétés suivantes par rapport à la valeur cible $\frac{1}{\sqrt{n_0}}$ de l'antenne de départ Ant_0 (voir figure III.5) :

- pour les ERel non regroupés : la valeur cible $\frac{1}{\sqrt{n}}$ augmente,
- pour les ERel faisant partie d'un regroupement : la valeur cible $\frac{1}{\sqrt{4}\sqrt{n}}$ ou $\frac{1}{\sqrt{3}\sqrt{n}}$ ou $\frac{1}{\sqrt{2}\sqrt{n}}$ diminue.

Or, si nous calculons les alimentations optimales de Ant_0 en relâchant la contrainte sur les modules, dit **cas relaxé**, nous introduisons un **degré de liberté supplémentaire** par rapport au calcul qui respecte la contrainte, dit cas non relaxé : l'antenne va **vérifier le gabarit** avec plus de **marge** qu'auparavant (voir les performances de Ant_0 à la section 4.4.2 page 100 du chapitre précédent). Nous pouvons alors calculer une alimentation qui concentre plus de puissance dans le spot visé : bien qu'elle ne corresponde pas à une alimentation qui pourra être appliquée réellement à l'antenne, elle va être utile pour résoudre notre problème.

En effet, nous faisons apparaître une information supplémentaire pour chaque ERel : la valeur de module souhaitée pour obtenir la **meilleure performance sur le spot visé**.

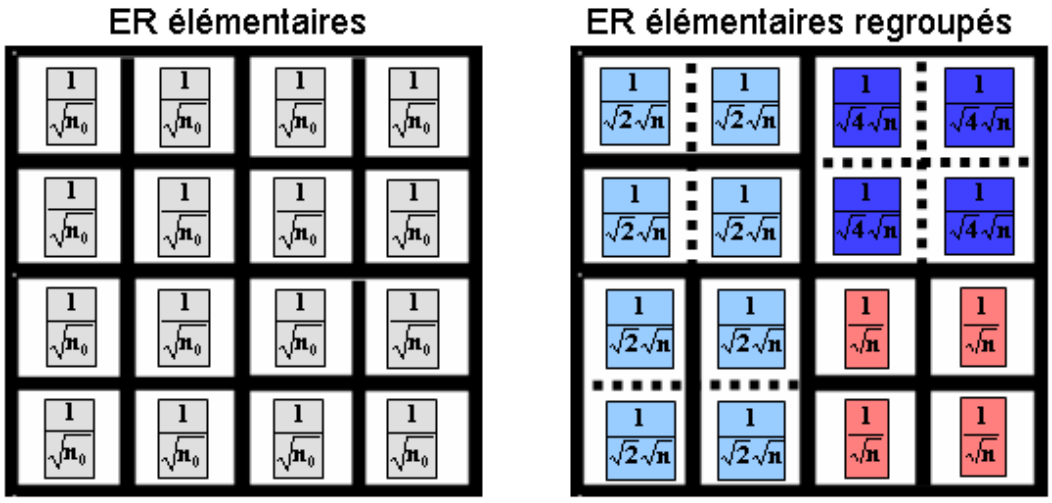


FIG. III.5 – Contrainte sur les modules : antenne initiale (à gauche) et antenne avec regroupements (à droite)

L'idée est alors d'effectuer les **regroupements d'ERel en fonction de cette information** : nous allons **rapprocher les valeurs cibles** de Ant_k des **valeurs souhaitées données par l'alimentation optimale dans le cas relaxé** pour Ant_0.

Ainsi, les performance de Ant_k seront dégradées car on dispose de moins de degrés de liberté dans l'espace des phases à cause des regroupements des ERel. Mais cette dégradation sera compensée dans le cas non relaxé : les valeurs cibles seront plus proches des valeurs de modules souhaitées pour les meilleures performances, et cette fois grâce aux regroupements, la contrainte sur les modules sera plus facile à respectée.

D'après ce qui précède, pour un ERel de Ant_0 :

- si la valeur de module souhaitée est supérieure à $\frac{1}{\sqrt{n_0}}$, on le laisse non regroupé dans Ant_k, pour faire augmenter sa valeur cible $\frac{1}{\sqrt{n}}$,
- si la valeur de module souhaitée est inférieure à $\frac{1}{\sqrt{n_0}}$, on essaie de regrouper cet ERel dans Ant_k , pour faire diminuer sa valeur cible $\frac{1}{\sqrt{r_i}\sqrt{n}}$, $1 \leqslant r_i \leqslant 4$.

3.2 Idée clé : technique de réduction de données

3.2.1 Introduction

Il reste un obstacle pour expoiter l'idée précédente : la valeur de module souhaitée pour un ERel donné n'est pas la même suivant le spot choisi pour le calcul de l'alimentation optimale dans le cas relaxé. Il nous faut donc une **méthode de réduction de données** pour obtenir la valeur qui soit le meilleur compromis entre les valeurs trouvées sur l'ensemble des spots : nous avons choisi d'utiliser la **décomposition en valeurs singulières** (**SVD** pour *Singular Value Decomposition* dans la suite).

3.2.2 Décomposition en valeurs singulières (SVD)

La méthode SVD permet de décomposer une matrice A sous la forme :

$$A = USV^* \tag{III.11}$$

avec

- S est une matrice diagonale carrée telle que

$$S = \begin{pmatrix} \sigma_1 & & (0) \\ & \ddots & \\ (0) & & \sigma_s \end{pmatrix} \qquad \in \mathbb{R}^{s\times s}. \tag{III.12}$$

  - Les éléments diagonaux sont les valeurs singulières $\{\sigma_j,\ j = 1 \dots s\}$ de A, positives et rangées par ordre décroissant.

    Généralement, elles décroissent très vite.
  - $\{\sigma_j^2,\ j = 1 \dots s\}$ est le spectre de AA^*.

- U est une matrice rectangulaire telle que

$$U = \begin{pmatrix} & & \\ U_1 & \dots & U_s \\ & & \end{pmatrix} \qquad \in \mathbb{R}^{n\times s}. \tag{III.13}$$

  - U est une matrice unitaire : $UU^* = I_n$ (idem pour V) : les vecteurs colonnes $\{U_j,\ j = 1 \dots s\}$ forment une base orthonormée.
  - $\{U_j,\ j = 1 \dots s\}$ sont les vecteurs propres de AA^*.

La décomposition de AA^* dans la base de vecteurs propres $\{U_j,\ j = 1 \dots s\}$ est $AA^* = (USV^*)(VS^*U^*) = USS^*U^*$ car V est unitaire, c'est-à-dire

$$AA^* = U \begin{pmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_s^2 \end{pmatrix} U^*. \tag{III.14}$$

On peut alors reconstruire A à partir de U :

- En effet, on écrit A comme somme de matrices de rang 1 d'après le produit $A = USV^*$

$$A = \sum_{j=1}^{s} \sigma_j\ U_j\ V_j^* \tag{III.15}$$

  avec U_j et V_j, $j = 1 \dots s$ les vecteurs colonnes de U et V.

- On peut alors approcher A par en prenant un nombre r petit devant s de valeurs singulières et vecteurs de U correspondants. La matrice reconstruite $\widetilde{A}$ est :

$$\widetilde{A} := \sum_{j=1}^{r} \sigma_j \; U_j \; V_j^* \quad \text{avec } r \ll s. \tag{III.16}$$

- On peut montrer (voir [GVL96]) que l'erreur entre la matrice initiale et la matrice reconstruite s'écrit avec les valeurs singulières négligées :

$$\|A - \widetilde{A}\|^2 = \sigma_{r+1}^2 + \ldots + \sigma_s^2$$

avec $\|.\|$ la norme de Frobenius.

Ainsi, l'approximation $A \backsimeq \widetilde{A}$ est justifiée car si les valeurs singulières décroissent vite, l'erreur précédente est négligeable.

Plus une valeur singulière est grande, plus le vecteur colonne de U correspondant participe à la reconstruction de A.

3.2.3 Exemple d'utilisation de la SVD avec une image

L'exemple suivant illustre comment l'information stockée dans une matrice peut être contenue dans très peu de vecteurs.

- On construit une matrice rectangulaire $A_{ex} \in \mathbb{R}^{1600\times100}$, avec la représentation suivante en niveaux de couleurs.
- On effectue la SVD sur A_{ex}, on obtient U_{ex} et V_{ex}.

  La décroissance des valeurs singulières est très rapide : la troisième est déjà négligeable.
- On représente la matrice reconstruite avec la première valeur singulière :

$$A_{rec1} = \sigma_1 \; U_{ex}^1 \; V_{ex1}^*. \tag{III.17}$$

- On représente la matrice reconstruite avec les deux premières valeurs singulières :

$$A_{rec2} = \sigma_1 \; U_{ex}^1 \; V_{ex1}^* + \sigma_2 \; U_{ex}^2 \; V_{ex2}^*. \tag{III.18}$$

  Les deux premières valeurs singulières nous ont permis de reconstruire A : les deux premiers vecteurs colonnes de U_{ex} contiennent l'essentiel de l'information stockée dans A_{ex}.

3.2.4 Mise en oeuvre : matrice des alimentations

Après avoir calculé les alimentations optimales de Ant_0 dans le cas relaxé pour l'ensemble des s spots, on range en colonnes les modules et on obtient une matrice A de taille $n_0 \times s$.

On applique la SVD à cette matrice, on espère une décroissance très rapide des valeurs singulières afin d'utiliser dans le meilleur des cas un seul vecteur propre de taille n_0, qui donnera alors une seule valeur de module souhaitée pour chaque ERel.

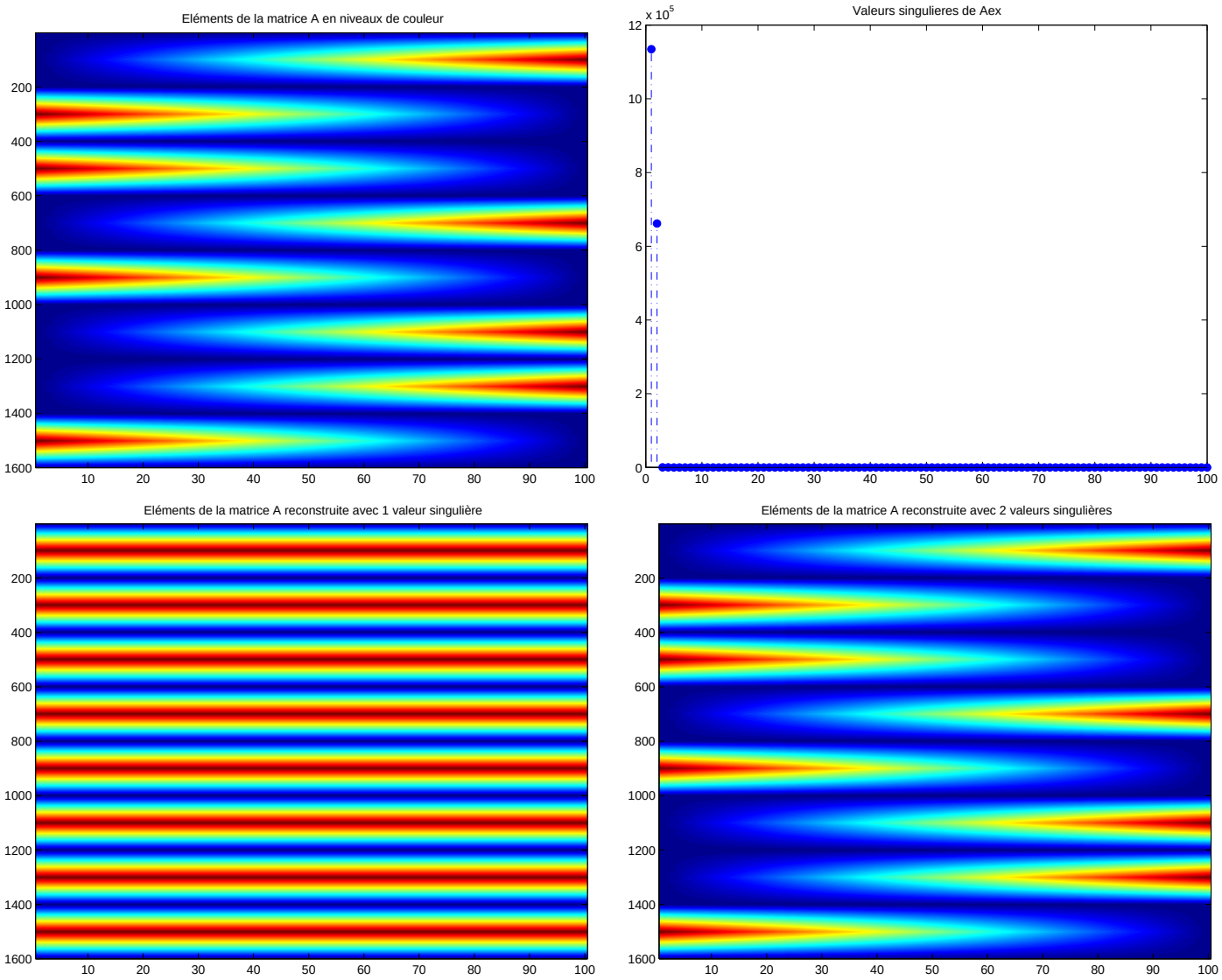


FIG. III.6 – Exemple d'utilisation de la SVD :
En haut : matrice de départ A_{ex} (à gauche) et valeurs singulières (à droite)
En bas : matrices reconstruites A_{rec1} (à gauche) et A_{rec2} (à droite)

3.3 Fonction coût

L'antenne de départ Ant_0 est constituée d'un réseau régulier de n_0 d'ERel, et selon la topologie choisie en définissant des regroupements à l'itération k, nous allons obtenir des antennes successives Ant_k.

Pour une antenne regroupée Ant_k formée de n ER, nous avons pour les indices d'ERel sous-jacents $i \in \{1 \dots n_0\}$ de l'alimentation sur les ERel :

- les valeurs cibles, notées $c_i^n = \frac{1}{\sqrt{r_i}\sqrt{n}}$, avec $1 \leqslant r_i \leqslant 4$ le nombre d'ERel qui forment l'ER regroupé auquel appartient l'ERel d'indice i.

  Elles varient suivant les regroupements effectués et le nombre total d'ER n,

- les valeurs souhaitées, notées m_i, issues de la SVD.

Elles sont fixes car elles sont issues des calculs effectués avec Ant_0.

Nous construisons une **fonction coût** à minimiser qui mesure pour Ant_k **l'erreur entre les valeurs cibles et les valeurs de module souhaitées** sur les ERel sous-jacents :

$$\begin{aligned} J_k &= \|(c_1^n - m_1, \ldots, c_0^n - m_0)\|^2 \\ &= \sum_{i=1}^{n_0} (c_i^n - m_i)^2. \end{aligned} \tag{III.19}$$

La valeur de départ est J_0 calculée avec Ant_0, pour la faire diminuer il faut effectuer des regroupements.

Si la valeur de J_k est proche de zéro, les valeurs cibles de Ant_k sont proches des valeurs souhaitées, et le calcul des alimentations optimales dans le cas non relaxé sera facilité au niveau des modules avec un nombre total d'ER qui aura diminué.

4 Construction de l'antenne optimisée

4.1 Algorithme de choix de la topologie initiale

Lors des tests, nous avons obtenu un meilleur résultat en initialisant l'antenne avec une géométrie qui présente des regroupements qui diminuent la fonction coût. On peut considérer que le but est de trouver un minimum global de la fonction coût J, mais on ne sait pas s'il est unique : on choisit alors un point de départ de l'algorithme (c'est-à-dire une géométrie d'antenne) qu'on pense proche de la solution qu'on veut obtenir. Ainsi, même si l'algorithme trouve un minimum local, la solution sera satisfaisante.

Soit Ant_0 l'antenne de départ composée d'un réseau de n_0 ERel, pour chaque taille $k = 1, \ldots, 4$ d'ER, on commence par définir les zones d'ERel notées Z_k susceptibles de convenir pour les regroupements de taille k.
Pour cela, nous allons estimer le nombre total d'ERel n_1 de l'antenne Ant_1 que nous voulons obtenir. Nous avons ainsi pour les ER de taille k de Ant_1 la valeur cible $c_k = \dfrac{1}{\sqrt{k}\,\sqrt{n_1}}$ des ERel sous-jacents. Nous voulons pour l'antenne optimale cette valeur cible le plus proche possible de la valeur de module souhaitée issue de la SVD des lois optimales pour Ant_0. Nous choisissons alors un intervalle de valeurs centré autour de c_k, et nous définissons Z_k comme l'ensemble des ERel de Ant_0, dont la valeur de module souhaitée est dans cet intervalle.

Pour obtenir une partition des ERel de Ant_0, nous choisissons les intervalles autour des valeurs c_k de façon à avoir une partition de l'intervalle des valeurs souhaitées.

L'algorithme pour le choix des zones d'ERel candidates Z_k aux regroupements de taille k est présenté dans le tableau III.1.

Nous avons besoin à présent de choisir une géométrie à partir des zones de regroupement candidates Z_k afin de définir Ant_1.

Algorithme 1-a : Calcul de la zone candidate Z_k

▷ **Entrées**

- Ant_0 antenne de départ à n_0 ERel
- $(m_1, \ldots, m_{n_0}) \in \mathbb{C}^{n_0}$ le vecteur des valeurs de module souhaitées issues de la SVD
- $k \in \{2, \ldots, 4\}$ les tailles d'ER regroupés pour l'antenne finale

▷ **Initialisation**

- On pose $n_1 = \frac{2}{3}\, n_0$ l'estimation du nombre d'ER de l'antenne initiale Ant_1 qu'on veut construire
- Soit $c_k = \dfrac{1}{\sqrt{k}\,\sqrt{n_1}}$, $k \in \{1, \ldots, 4\}$ les valeurs cibles pour Ant_1
- Soit $[u_m, u_M]$ l'intervalle des valeurs de modules souhaitées, avec $u_m := \min\limits_{i=1\ldots n_0} m_i$ et $u_M := \max\limits_{i=1\ldots n_0} m_i$

▷ **Choix des zones**

- Découper l'intervalle $[u_m, u_M]$ en une partition de 4 intervalles I_k, chacun contenant une des valeurs cibles c_k, $k \in \{1, \ldots, 4\}$
- Pour chaque intervalle I_k, $k \in \{2, \ldots, 4\}$,
  associer une zone d'ERel Z_k dont les valeurs de modules souhaitées appartiennent à I_k.
  Ces ERel constituent les candidats à un regroupement de taille k (voir figure III.10 page 125) et ces zones définissent une partition de l'ensemble des ERel de l'antenne de départ.

TAB. III.1 – Algorithme de calcul de la zone candidate

Pour une taille de regroupement $k \in \{2, \ldots, 4\}$ et une zone Z_k fixée, nous devons d'abord travailler dans les zones connexes de Z_k dans lesquelles nous pouvons emboîter les ER regroupés en laissant le moins possible d'ERel non regroupés.

Comme les zones connexes comportent un nombre faible d'ERel (voir figure III.10 page 125), nous pouvons essayer toutes les combinaisons de regroupements possibles dans chacune d'entre elles (on utilise pour cela une boucle récursive) et choisir la géométrie qui maximise le nombre de regroupements, c'est-à-dire qui minimise le nombre d'ER.
Nous obtenons Ant_1 en appliquant tous les regroupements choisis à l'étape précédente.

L'algorithme pour le calcul de l'antenne initiale nommée Ant_1 est alors présenté dans le tableau III.2.

Algorithme 1 : choix de la topologie initiale

▷ **Entrées**

- Ant_0 antenne de départ à n_0 ERel
- $U \in \mathbb{C}^{n_0}$ vecteur des valeurs de module souhaitées issues de la SVD
- $k = \{2, \ldots, 4\}$ les tailles d'ER regroupés pour l'antenne finale

▷ **Choix des regroupements**

- Pour $k = 2, \ldots, 4$ les tailles de regroupements possibles
  - **Algorithme 1-a** : calcul de la zone Z_k des ERel candidats aux regroupements de taille k
  - Calcul des N_k zones connexes $Z_{k,j}$, $j \in \{1 \ldots N_k\}$ composant Z_k
  - Pour $j = 1 \ldots N_k$ chaque zone connexe
    - ▷ **Calcul de tous les regroupement de taille k dans $Z_{k,j}$**
      - Soit L la liste des ERel constituant $Z_{k,,j}$
      - **Boucle récursive :** Si L n'est pas vide
        - Soit le plus petit indice d'ERel $i_{ER} \in L$
        - Effectuer tous les regroupements possibles de taille k avec les ERel voisins de i_{ER}
        - Pour chaque regroupement, soit J la liste d'ERel le définissant
          - $L = L \setminus J$
          - **Récursivité** : retour au début de la boucle parcourant L

        Sinon conserver la géométrie obtenue sur $Z_{k,j}$.
      - Choisir la géométrie précédente qui minimise le nombre d'ER dans $Z_{k,j}$

▷ **Sortie**

– Ant_1 est la géométrie obtenue avec les regroupements choisis dans la boucle précédente.

TAB. III.2 – Algorithme de choix de la topologie initiale

4.2 Algorithme d'optimisation topologique

Nous partons de l'antenne initiale Ant_1, et pour construire l'antenne finale optimisée, nous voulons diminuer l'écart ou erreur entre les valeurs cibles et les valeurs de modules souhaitées issues de la SVD. Nous utilisons pour cela la fonction coût J_k définie en section 3.3 page 115 qui somme les erreurs au carré sur tous les ERel sous-jacents de Ant_1.

L'espace de recherche pour cette fonction est l'ensemble des géométries admissibles de regroupements des ERel de Ant_0, avec les contraintes de taille, de forme et de géométrie pour la juxtaposition des ER regroupés.

Nous utilisons un algorithme inspiré par la méthode du gradient topologique pour minimiser la fonction coût : soit l'ERel qui correspond à l'erreur la plus grande, nous avons intérêt à modifier la topologie à cet endroit là. Ici, au lieu de faire un trou comme dans la méthode du gradient toplogique, nous effectuons un regroupement : il ne reste plus qu'à choisir le meilleur regroupement avec les ERel voisins.

L'algorithme pour construire l'antenne optimisée est présenté dans le tableau III.3.

4.3 Validation

Notre méthode induit qu'il faut valider *a posteriori* les performances de l'antenne obtenue Ant_{opt}.

On applique alors pour l'ensemble des spots l'algorithme de recherche des lois optimales dans le cas non relaxé des contraintes de module, et on vérifie que le gabarit est toujours respecté.

Algorithme 2 : optimisation topologique

▷ **Entrées**

- Ant_1 l'antenne issue de l'algorithme 1 : choix de la topologie initiale
- $U \in \mathbb{C}^{n_0}$ vecteur des valeurs de module souhaitées issues de la SVD
- $k = \{2, \ldots, 4\}$ les tailles d'ER regroupés pour l'antenne finale
- les formes d'ER regroupés admissibles

▷ **Initialisation**

- $Ant_k = Ant_1$
- $I_k = I_0 = 1, \ldots, n$ la liste des ERel de l'antenne initiale
- Calculer la fonction coût $J_k = \sum_{i=1}^{n_0} (c_i^n - m_i)^2 = J_0$

▷ **Boucle d'optimisation**

- Tant que J_k diminue
  - Dans la liste I_k, calculer l'indice d'ERel i_M pour lequel l'erreur $(c_i^n - m_i)^2$ est maximale
  - Effectuer tous les regroupements admissibles de l'ERel i_M avec ses ERel voisins : on s'autorise à casser les regroupements existants pour réaliser le regroupement à tester.
    Pour chaque nouvelle antenne $Ant_{\tilde{k}}$ obtenue,
    calculer la valeur de $J_{\tilde{k}}$ associée
  - Choisir la nouvelle topologie Ant_k parmi les précédentes pour que J_k soit minimale
  - $I_k = I_k \setminus i_M$

▷ **Sortie**

- $Ant_{opt} = Ant_k$

TAB. III.3 – Algorithme d'optimisation topologique

5 Etude industrielle

5.1 Matrice des alimentations

Nous reprenons l'étude OTOP avec les résultats d'optimisation des lois d'alimentation dans le cas relaxé (voir section 4.4.2 du chapitre II page 100) pour l'antenne de départ Ant_0 à 529 ERel (voir figure II.21 dans la section 4.2.2 page 87).

Nous formons une matrice A de taille 529×44 en rangeant en colonnes les modules des alimentations optimales pour l'ensemble des 44 spots.

5.2 Résultat de la SVD

5.2.1 Valeurs singulières

Nous effectuons la SVD sur la matrice A, nous obtenons les matrices S et U.

Nous affichons dans la figure III.7 la décroissance des valeurs singulières obtenues.

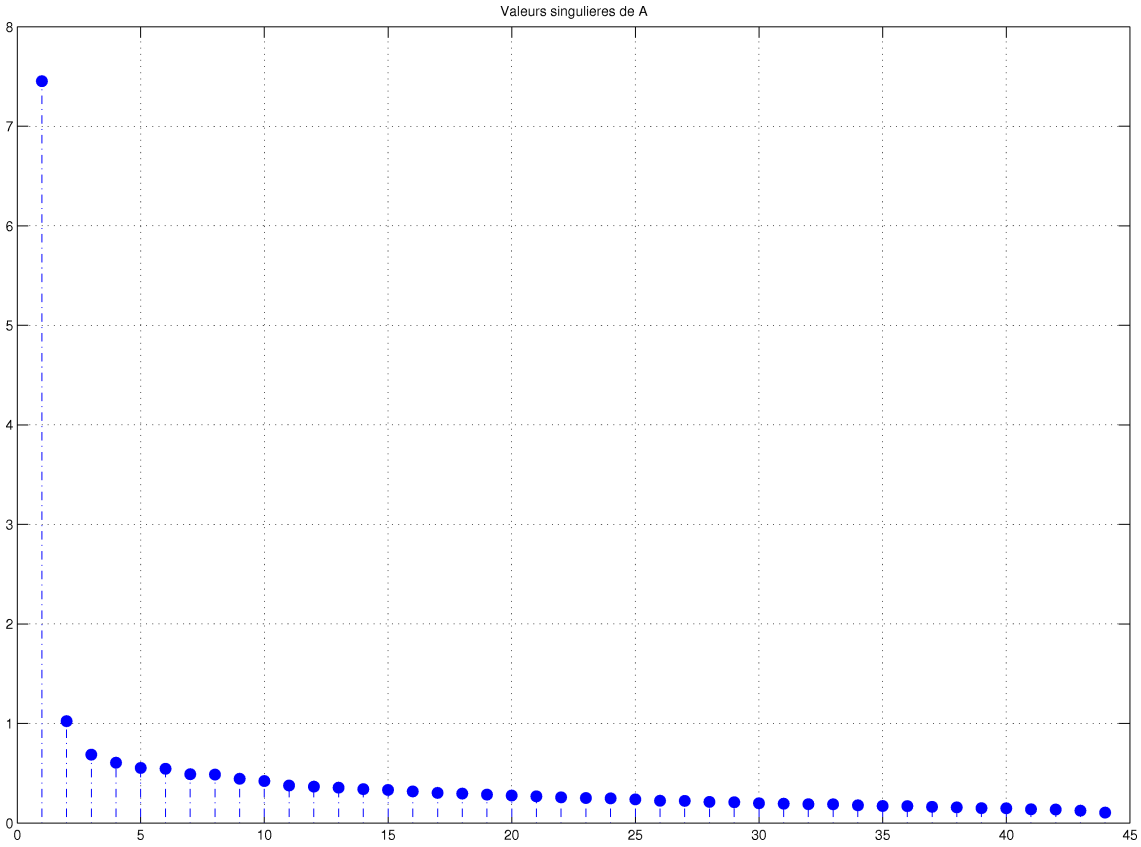


FIG. III.7 – Valeurs singulières de la matrice des modules des alimentations optimales relaxées de Ant_0

Comme on s'y attendait, la première valeur singulière est prépondérante par rapport aux suivantes, donc le premier vecteur de la base définie par les colonnes de U contient l'essentiel de l'information sur les n_0 valeurs de module optimales pour l'ensemble des spots.

5.2.2 Carte des modules souhaités

Nous utilisons donc le premier vecteur colonne de la matrice U qui donne les valeurs de modules souhaitées pour obtenir les meilleures performances sur tous les spots, et la carte obtenue est affichée dans la figure III.8.

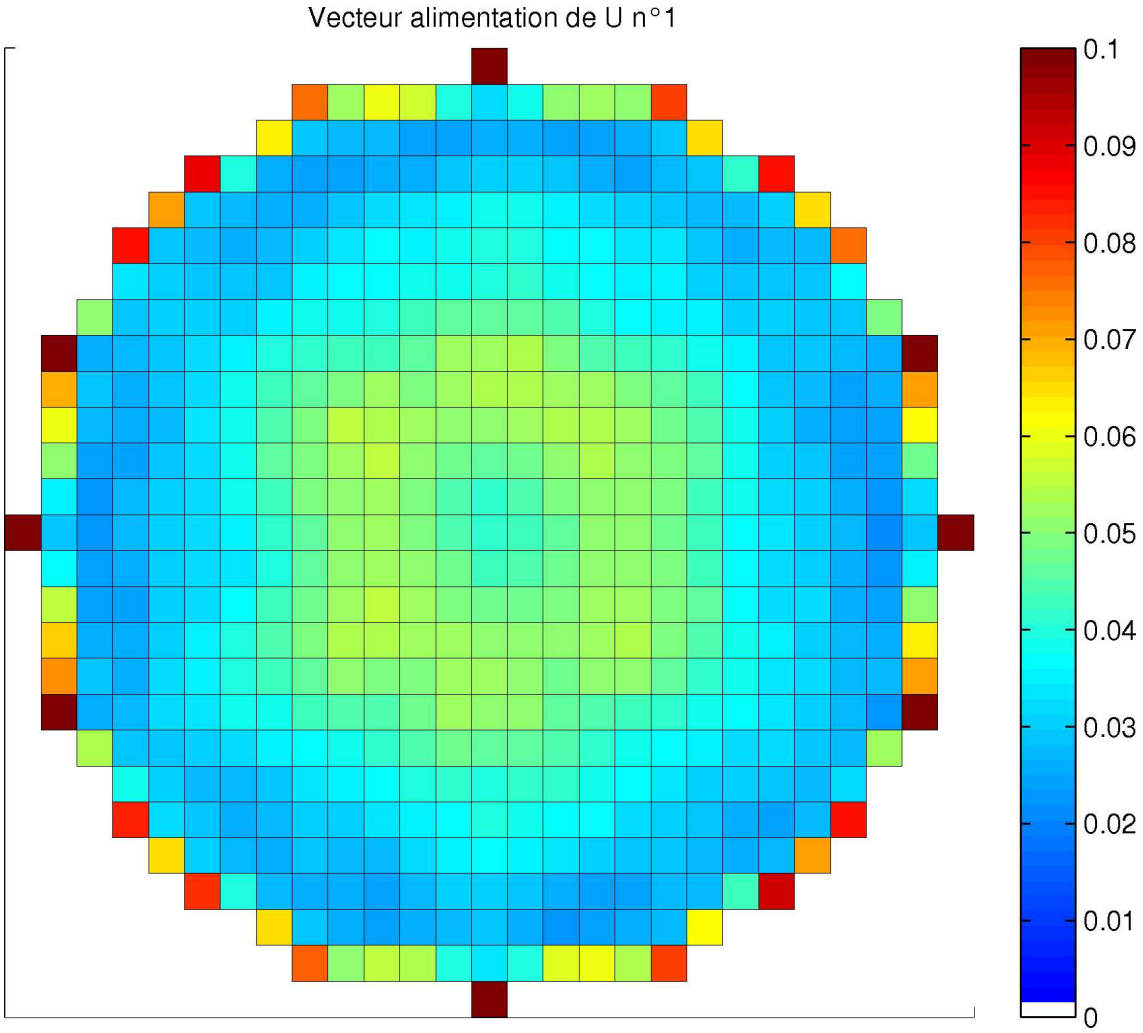


FIG. III.8 – Carte de module : premier vecteur colonne de U

5.2.3 Interprétation

Quand on observe la carte des modules souhaités précédente, on retrouve la tendance qu'on pouvait visualiser sur les cartes des modules des alimentations optimales dans le cas relaxé :

- des valeurs fortes au centre et sur les bords de l'antenne,
- une ou deux couronnes successives autour du centre avec des valeurs qui diminuent.

L'algorithme va donc essayer de regrouper des ERel dans les couronnes où les valeurs de module souhaitées sont faibles.

5.3 Contraintes géométriques sur les ER

Les motifs du panel présenté en section 2.1 page 106 provoquent des difficultés géométriques lors de la juxtaposition des ER. Afin de limiter cet inconvénient, et connaissant la forme de la zone où s'effectuerons les regroupements (une couronne), nous avons choisi de partager l'antenne en neuf zones et de réduire la liste des ER admissibles par zone.

Dans le panel de motifs, nous avons défini les formes réalisables industriellement. Ici, nous choisissons cette fois par région la forme et l'orientation des ER admissibles afin de pouvoir former des arcs de cercle plus facilement : voir la figure III.9 pour les ER admissibles.

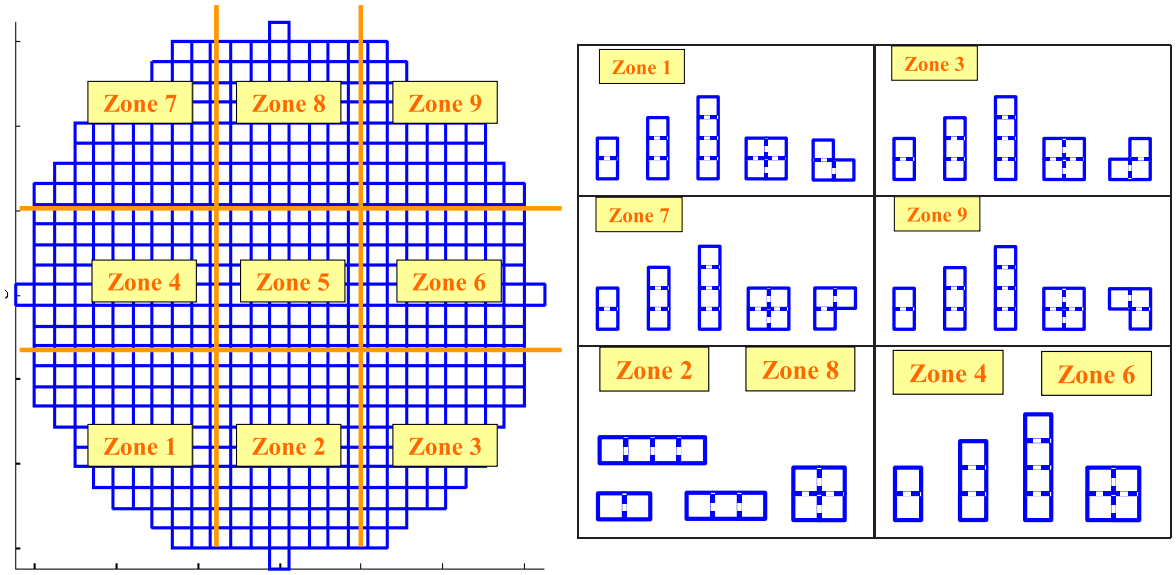


FIG. III.9 – ER admissibles et zones de l'antenne associées

5.4 Antenne optimale

5.4.1 Antenne initiale Ant_1

Afin de trouver une antenne initiale pour l'algorithme d'optimisation (voir section 4.1 page 116), à partir de la carte des modules souhaités nous définissons des intervalles de valeurs de modules et des zones d'ERel candidats aux regroupements associées : voir les figures III.10 et III.11.

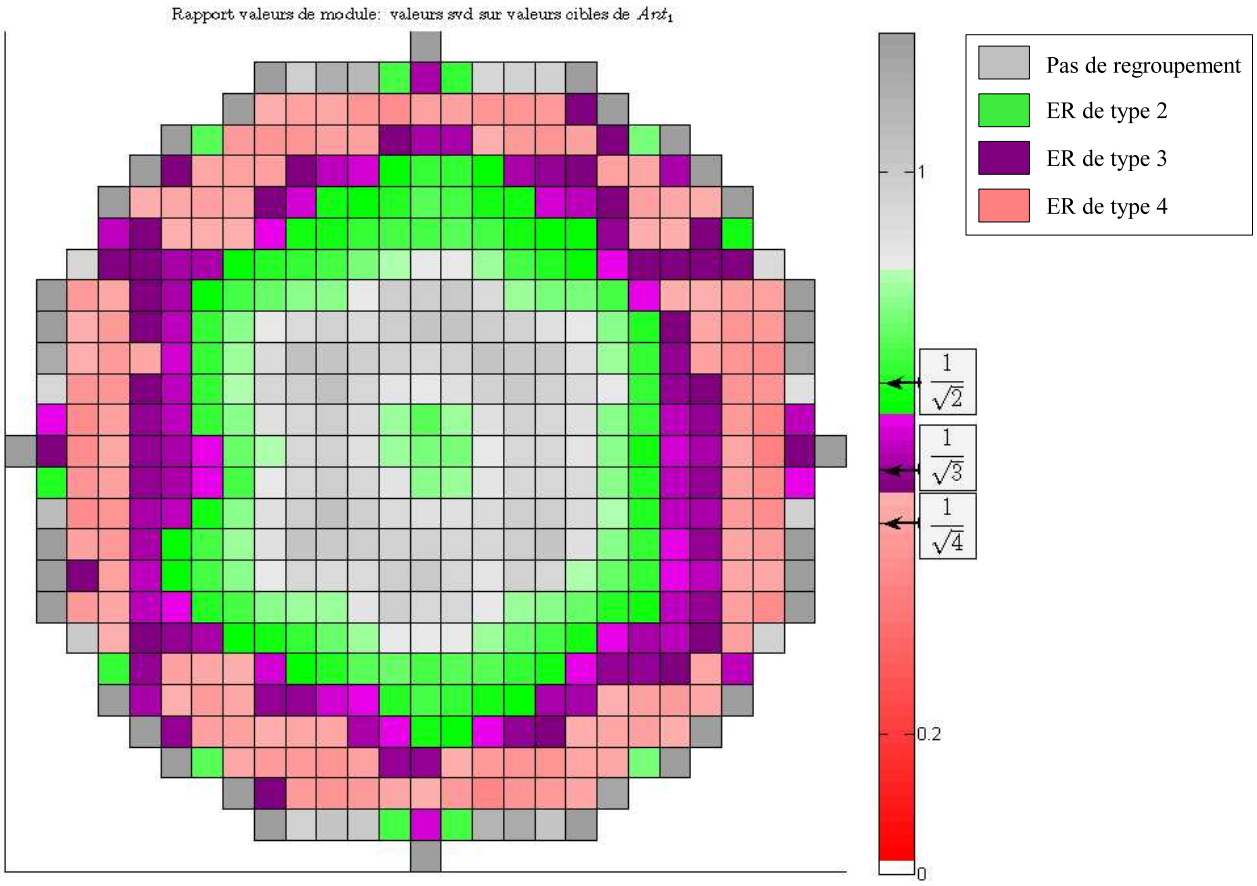


FIG. III.10 – Carte présentant le rapport entre les valeurs de modules souhaitées issues de la SVD et les valeurs cibles de l'antenne objectif initiale Ant_1 : intervalles pour la définition des zones Z_k , $k \in \{1, \ldots, 4\}$

Nous appliquons l'algorithme 1 (voir page 119), et nous obtenons l'antenne initiale Ant_1 à 390 ER : voir figure III.12.

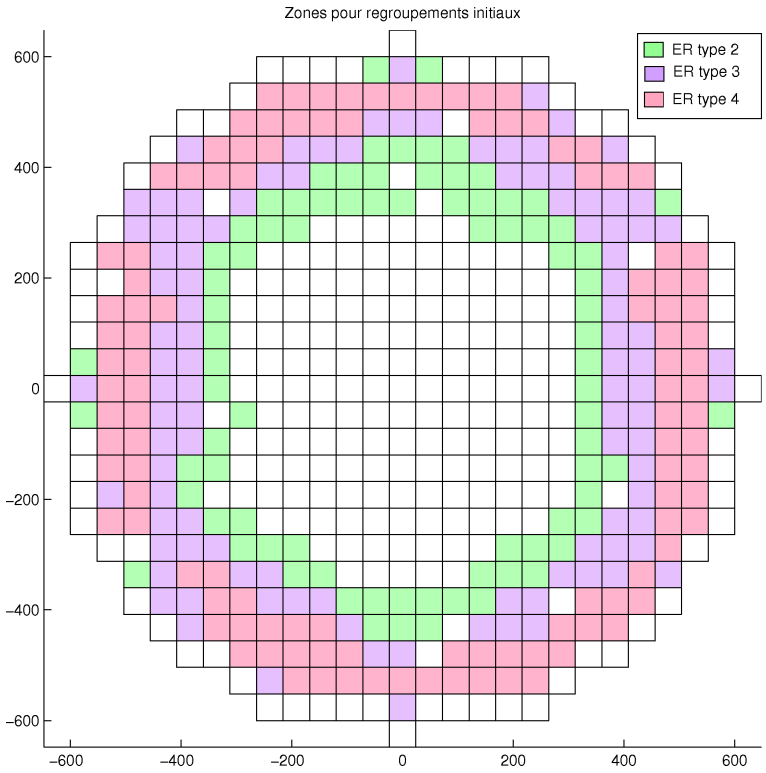


FIG. III.11 – Zones candidates aux regroupements de taille 2, 3 ou 4

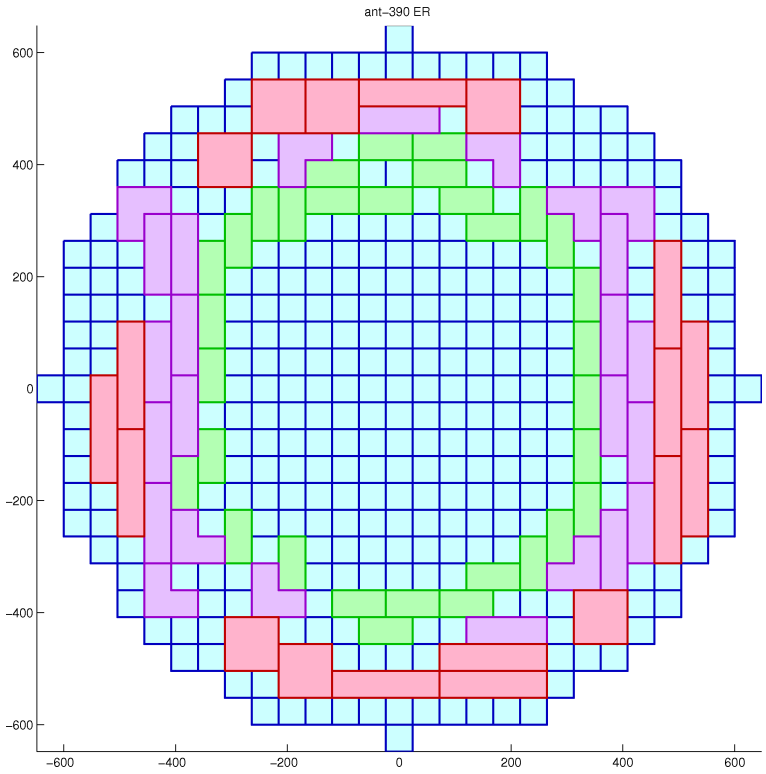


FIG. III.12 – Antenne inititiale Ant_1 : 390 ER

5.4.2 Résultat de l'algorithme d'optimisation topologique : Ant_{opt}

A présent, nous appliquons l'optimisation topologique, c'est-à-dire l'algorithme 2 (voir page 121), avec l'antenne initiale Ant_1 comme point de départ.

La décroissance de la fonction coût J, qui mesure l'écart aux valeurs de modules souhaitées, est présentée dans la figure III.13, avec les valeurs suivantes :

- $J_0 = 0.1417$ pour Ant_0,
- $J_1 = 0.1038$ pour Ant_1 (point de départ, itération 0),
- J_k pour Ant_k les antennes successives trouvées au cours des itérations suivantes,
- $J_{opt} = 0.0612$ pour Ant_{opt} l'antenne optimisée.

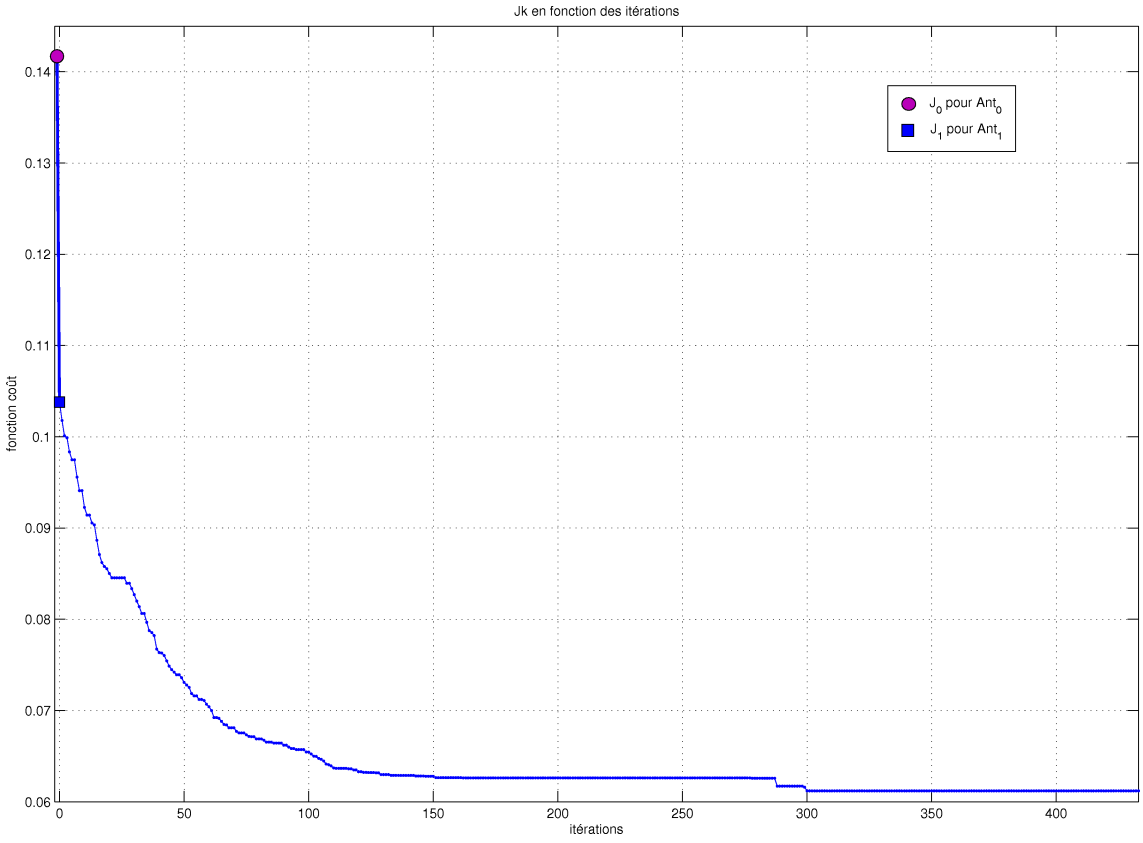


FIG. III.13 – Décroissance de la fonction coût J

L'antenne Ant_1 a été construite pour se rapprocher de la topologie qui minimise J, et la valeur de J_1 (73% de J_0) est bien inférieure à celle de J_0 .
L'algorithme fait ensuite diminuer la valeur de J jusqu'à J_{opt} (43% de J_0).

L'algorithme parcourt successivement tous les indices d'ERel dont la valeur de module souhaitée est inférieure à la valeur cible de départ : les 100 premières itérations permettent d'effectuer chacune un regroupement qui fait diminuer J. Mais sur les itérations restantes, on teste des indices d'ERel qui sont sont déjà regroupés ou voisins de regroupements et il est plus difficile de trouver une solution qui diminue J :

la courbe de décroissance est alors presque horizontale et la géométrie n'est pratiquement plus modifiée.

L'antenne optimale Ant_{opt} obtenue à la fin de l'algorithme d'optimisation topologique est présentée dans la figure III.14 : ellle est constituée de 284 ER, ce qui réprésente 54% du nombre d'ER de l'antenne de départ.

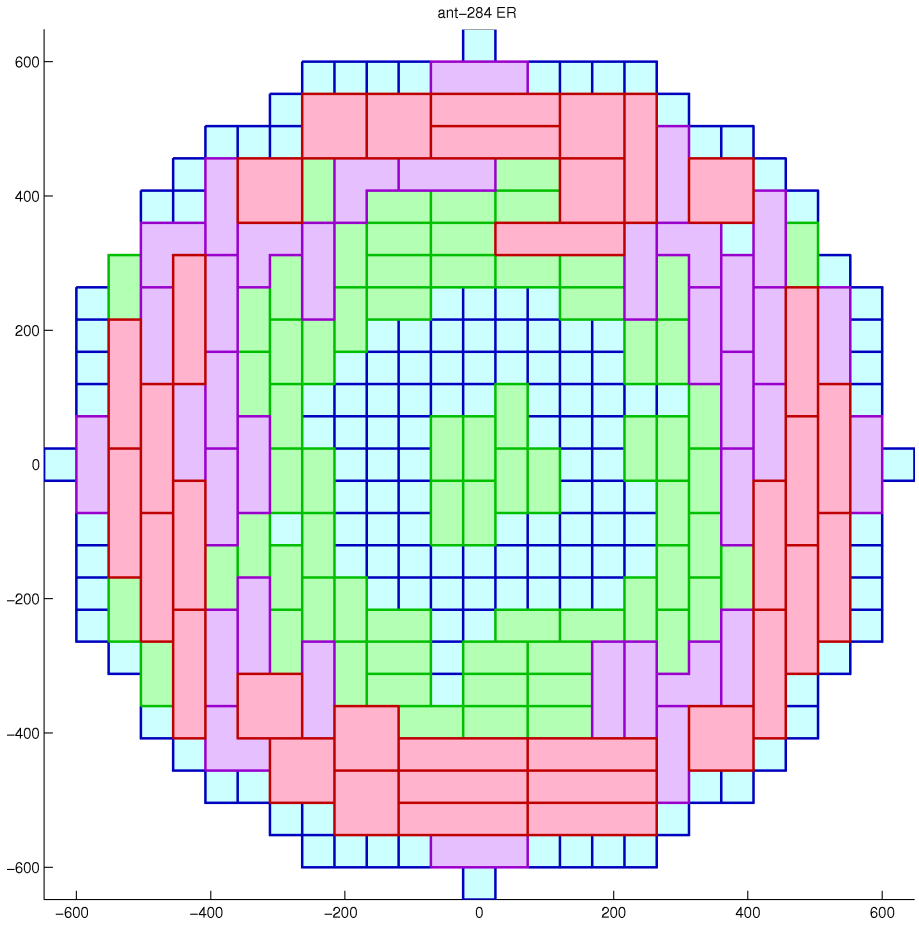


FIG. III.14 – Antenne optimale Ant_{opt} : 284 ER

5.4.3 Amélioration : Ant_{opt-b}

Heuristique : regroupements sur le bord A la présentation de l'antenne optimisée, les ingénieurs de recherche de Thalès ont proposé de diminuer encore le nombre d'ER à l'aide d'une heuristique : en effet, l'expérience de construction de ce type d'antennes montre que la densité de sources est décroissante du centre vers le bord. Or sur notre solution, une couronne d'ERel sur le bord extérieur de l'antenne n'a pas été regroupée, conformément aux informations issues de la SVD. Cela signifie que l'on doit pouvoir regrouper ces ERel avec les ER regoupés voisins sans dégrader les performances de l'antenne.

Pour intégrer cette information à l'algorithme d'optimisation topologique, il faudrait modifier l'algorithme de recherche des lois optimales afin de contrôler et contraindre

le rayonnement d'une manière différente et qu'en conséquence les valeurs de modules calculées dans le cas relaxé ne soit pas aussi fortes sur le bord extérieur de l'antenne.

Dans notre cas, cette heuristique va avoir pour effet de diminuer le nombre total d'ER mais il ne faut plus calculer la fonction coût J associée puisqu'on n'utilise plus les valeurs de modules issues de la SVD dans cette partie, et J va au contraire augmenter.

Le principe est le suivant : on regroupe les ERel de Ant_{opt} situés sur le bord avec ses voisins (des ERel ou des ER déjà regroupés) de manière à obtenir des ER admissibles. On s'autorise à casser les regroupements existants.

L'**antenne finale** Ant_{opt-b} obtenue après les regroupements sur le bord est présentée dans la figure III.15 : elle est constituée de 250 ER, ce qui réprésente 47% du nombre d'ER de l'antenne de départ.

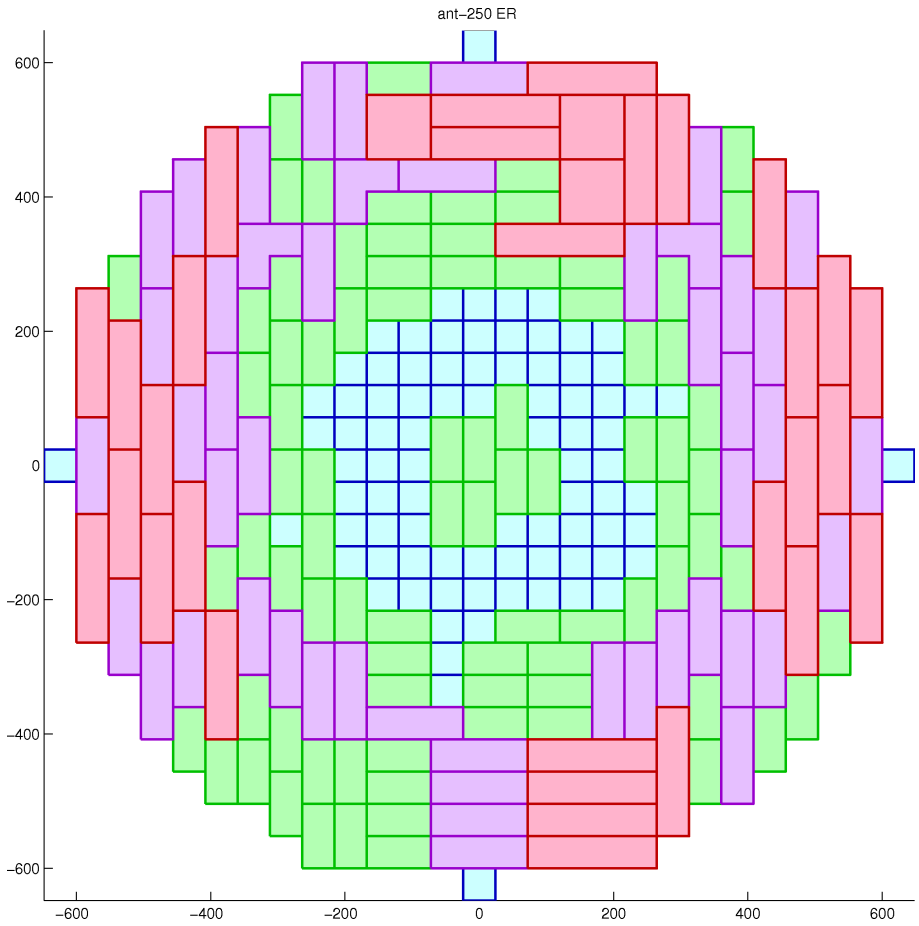


FIG. III.15 – Antenne optimale avec regroupements sur le bord Ant_{opt-b} : 250 ER

5.4.4 Validation

Performances Il faut à présent valider *a posteriori* les performances de l'antenne Ant_{opt-b} obtenue : nous utilisons l'algorithme d'optimisation des alimentations dans le cas non relaxé sur l'ensemble des spots.

Cependant il existe à la fin de l'algorithme un écart entre les valeurs de module calculées et les valeurs cibles car la norme de leur différence n'est pas diminuée exactement jusqu'à zéro.

Pour avoir une solution conforme aux exigences définies par l'utilisateur, nous rappelons l'alimentation finale dans le cas non relaxé projeté (voir section 4.3.7 page 96) :

- on impose comme modules les valeurs cibles correspondant à la géométrie de l'antenne optimisée,
- on garde les phases optimales issues du cas non relaxé.

Exemple Nous affichons les niveaux de directivité, et les cartes de module et de phase, pour l'alimentation optimale du spot n°1 dans les figures III.16 et III.17.

La carte des modules projetés ne comporte que les quatre valeurs cibles correspondant aux cas de regroupements (non regroupé, ou regroupé dans un ER de taille 2,3 ou 4), et la carte des phases présente des bandes parallèlles similaires à celles rencontrées pour Ant_0 dans le cas relaxé.

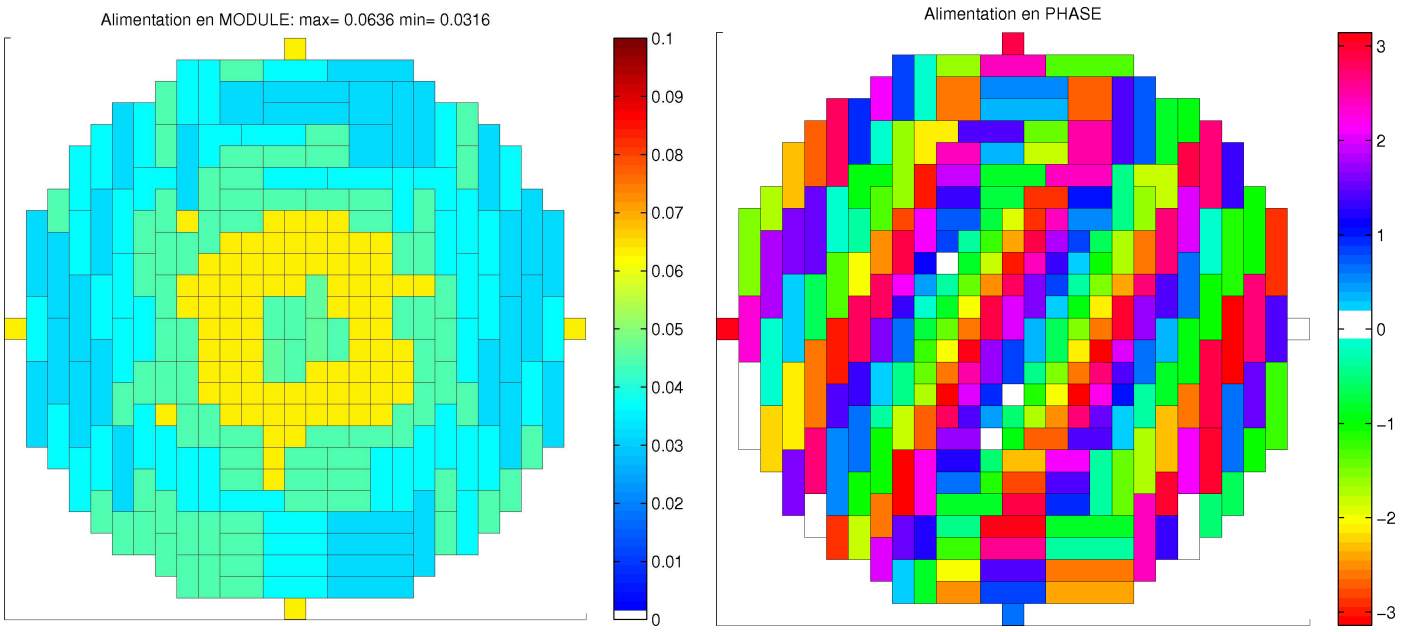


FIG. III.16 – Cartes de module et phase de l'alimentation optimale du spot n°1 Ant_{opt-b}, cas non relaxé

Analyse Les performances pour Ant_{opt-b} dans le cas non relaxé, puis dans le cas projeté sont présentées dans la figure III.18.

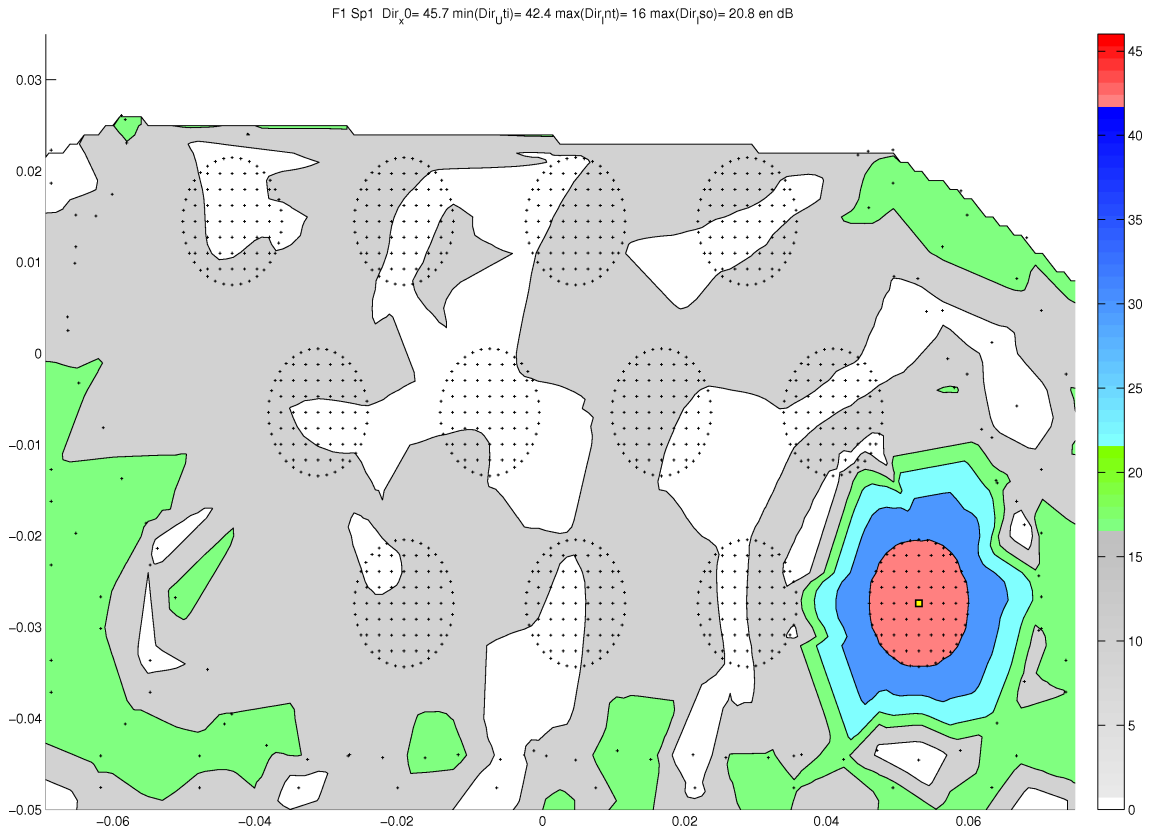


FIG. III.17 – Niveaux de directivité de l'alimentation optimale du spot n°1 : Ant_{opt-b}, cas non relaxé

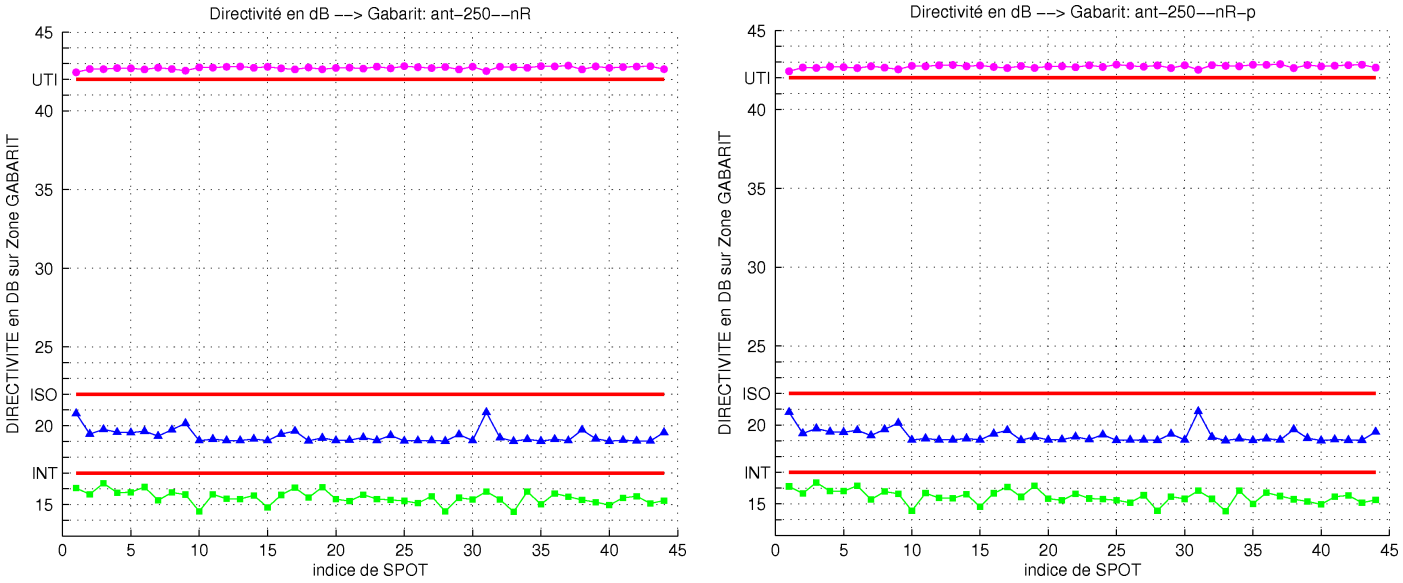


FIG. III.18 – Performances : antenne optimisée Ant_{opt-b}
A gauche : cas non relaxé
A droite : cas non relaxé projeté

Si nous comparons avec les performances de Ant_0 dans le cas relaxé (voir figure II.29 page 101), nous constatons que les valeurs sont légèrement dégradées sur les zones d'interférence et d'isolation, mais pour une antenne optimisée Ant_{opt-b} dont le nombre d'ER et donc de contrôles a été réduit de moitié, et pour laquelle on ne contrôle que les phases (au lieu des modules et des phases pour Ant_0).

5.4.5 Bilan

Pour constituer l'antenne finale optimisée Ant_{opt-b} (figure III.15), les **éléments rayonnants élémentaires de départ** ont été **regroupés en cinq types** seulement d'éléments (trois types rectangulaires un type carré et un type "coude"), formant un réseau irrégulier.

Le coût induit par la fabrication de ces nouveaux types d'éléments est largement compensé par les deux points suivants :

- le **nombre d'éléments a été réduit de moitié** (moins 53%) par rapport au nombre initial,
- les **performances de l'antenne finale** dans le **cas non relaxé** sont **équivalentes** à celles de l'**antenne de départ dans le cas relaxé**.

  Cela signifie :

  - dans le cas de l'antenne initiale, la structure de réseau régulier produit des lobes non désirés qui doivent être compensés par le contrôle des modules (cas relaxé) pour respecter les contraintes de rayonnement,
  - dans le cas de l'antenne optimisée, la structure de réseau irrégulier ne produit pas les lobes non désirés, et on peut se contenter de contrôler les phases (cas non relaxé) pour dépointer le faisceau.

  Ainsi, dans l'antenne finale respectant la contrainte sur les modules, un seul type d'amplificateur générant la même puissance (ou module) pour chaque élément rayonant (regroupé ou pas), sera fabriqué, ce qui représente une solution industrielle économique.

Conclusion

Bilan

Les algorithmes décrits dans ce document ont répondu avec succès au problème industriel posé par Thalès Alenia Space :

- La **méthode d'optimisation des lois d'alimentation**, avec ou sans contrôle des modules, pour une antenne réseau à **géométrie quelconque**, est **robuste et rapide** (20 itérations pour trouver l'optimum lors du cas test).

  Nous avons proposé une nouvelle technique permettant de **rendre le problème convexe** :

  - d'une part **en relaxant la contrainte d'égalité sur les modules** par une contrainte d'inégalité,
  - d'autre part en évitant de maximiser l'énergie à l'intérieur du spot. En effet, l'énergie est une fonction de type quadratique, et sa maximisation engendre de nombreux minima locaux.

    A cet effet nous avons

    - soit utilisé le principe de **conservation de l'énergie pour minimiser à l'extérieur du spot**,
    - soit **imposé une valeur du champ rayonné au centre du spot**.

- La **méthode d'optimisation topologique** originale que nous avons proposée a permis de concevoir sur un cas test une antenne optimale : elle a été obtenue par **regroupements des éléments rayonnants élémentaires** de départ, sur un **critère de modules optimaux** donnés par la méthode précédente.

  Le **nombre d'éléments** a été **réduit de moitié**, et l'antenne finale à réseau irrégulier présente des **performances de rayonnement similaires** à l'antenne initiale.

  De plus, ces performances sont atteintes avec le contrôle des phases uniquement : c'est un avantage industriel, car on peut fabriquer un seul type d'amplificateur délivrant une puissance unique pour chaque élément.

Création du logiciel OTOP

Cette étude a abouti à la création du logiciel OTOP (du nom du projet ANR retenu) sous MATLAB de *design* optimal d'antennes réseaux actives : la méthode

a été implémentée de manière à obtenir une optimisation topologique entièrement automatisée.

Le logiciel a été adapté aux normes spécifiées par Thalès par la société Ansys France.

Perspectives

La méthode doit être encore validée sur des cas tests représentatifs de couvertures multi-faisceaux typiques de futurs systèmes satellites.

Par exemple, la figure III.19 présente une couverture pour laquelle les spots ne sont plus circulaires mais dont la forme épouse les contours d'une zone linguistique.

L'algorithme d'optimisation des lois d'alimentation, basé sur l'hypothèse de spots fins et circulaires devra être adapté à cet effet.

On peut proposer une méthode de contrôle du champ E sur le bord d'un spot irrégulier :

$$\begin{cases} \displaystyle\sum_{j\in I_b} \lambda_j \; E(a, x_j) = d \\ \displaystyle\sum_{j\in I_b} \lambda_j = 1 \end{cases} \tag{III.20}$$

avec

- x_j, $j \in I_b$ les points sur le bord du spot,
- $d \in \mathbb{R}^+$ constante correspondant à la valeur de champ souhaitée sur le bord.

Une méthode de type Usawa permet de mettre à jour à chaque itération k les coefficents λ_j afin d'augmenter dans la somme $\sum_{j\in I_b} \lambda_j \; E(a, x_j)$ la contribution des points x_j pour lesquels la valeur de champ est la plus éloignée de la valeur souhaitée d (λ_j augmente quand $E(a^{(k)}, x_j) \leqslant d$) :

$$\begin{cases} \lambda_j^{(k+1)} := \lambda_j^{(k)} + \left(d - E(a^{(k)}, x_j)\right)^+ \\ \lambda_j^{(k+1)} = \dfrac{\lambda_j^{(k+1)}}{\sum_j \lambda_j^{(k+1)}}. \end{cases} \tag{III.21}$$

De plus, afin d'obtenir une réduction plus forte du nombre d'éléments, un axe de recherche pourrait être développé concernant l'algorithme topologique :

- on pourrait utiliser plus de formes admissibles et mieux prendre en compte les contraintes géométriques de regroupement, pour voir si on obtient un gain supplémentaire,
- une amélioration de la fonction coût pourrait inclure un terme supplémentaire permettant de minimiser le nombre total d'éléments, ou de minimimer le nombre de formes admissibles .

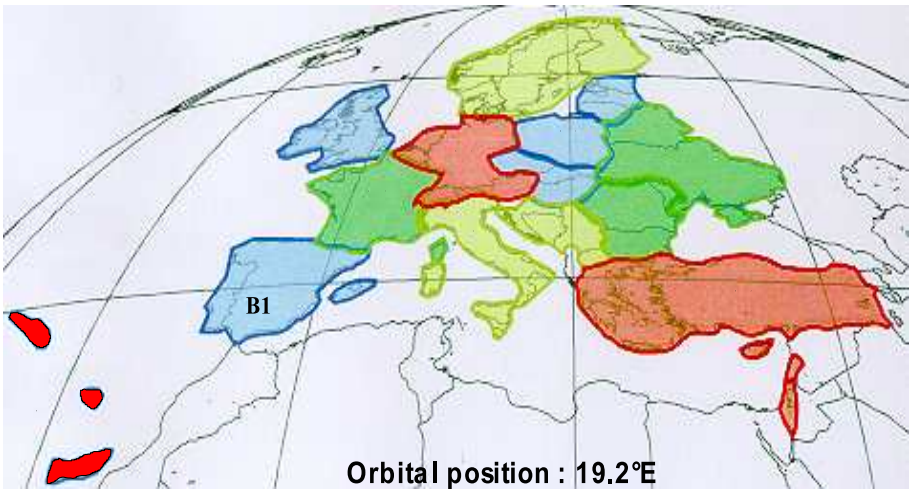


FIG. III.19 – Exemple de couverture linguistique de l'Europe

Deuxième partie

Etude de fiabilité et de sensibilité dans une modélisation d'accident chimique

Introduction

Dans de nombreux domaines de l'ingénierie, la simulation numérique a pris une place prépondérante. Au fur et à mesure de la progression des moyens informatiques de calcul et des méthodes numériques associées à la résolution des équations, les modèles numériques permettent de simuler les essais et les expériences, avec un gain considérable en temps et en coût.

On est aujourd'hui en mesure de modéliser des systèmes ou des phénomènes de plus en plus complexes, et l'enjeu issu de ces simulations devient de plus en plus important.

Ainsi se pose naturellement la question majeure de la qualité du modèle et de la validité des résultats, pour savoir dans quelle mesure on peut faire confiance à la réponse de la simulation.

L'exécution d'un modèle s'accompagne de plusieurs sources d'incertitudes et d'erreurs, et l'analyse d'incertitude consiste à s'efforcer de les quantifier.

On utilise à cet effet souvent des méthodes stochastiques, qui étudient les effets aléatoires par modélisation probabiliste.

La fiabilité, domaine de l'ingénieur, consiste plus généralement à évaluer les risques de manière à ce qu'ils restent acceptables.

En effet, toute prise de décision est accompagnée de l'acceptation d'un risque plus ou moins bien évalué. La gestion des incertitudes devient ainsi un outil d'aide à la décision en contribuant à mieux quantifier les risques.

Dans le cadre d'une collaboration avec le **CEG** (pour Centre d'Etudes de Gramat, de la DGA, pour Délégation Générale pour l'Armement), nous avons contribué à l'étude des conséquences d'un accident de type chimique : le **logiciel CEGAP** a été créé sur deux premiers cas tests à 9 et 30 paramètres incertains. Ce logiciel est en cours d'implémentation avec les logiciels de simulation du CEG, et sera amené à évoluer afin de traiter des modélisations de plus en plus réalistes.

La deuxième partie de ce manuscrit est organisée de la manière suivante :

- **Calculs de fiabilité** : nous exposons les sources d'incertitudes lors d'une modélisation numérique, et les approches possibles.

  Des lois de probabilité définissent les paramètres incertains, une fonction de modélisation G est définie, et nous devons répondre aux questions suivantes :

  - Quelle est la **probabilité** pour G de **dépasser un seuil donné** ?
  - Quelles sont les **conditions les plus probables** sur les paramètres qui permettent d'atteindre ce seuil ?

- Quels sont les **paramètres les plus influents** ?

Les hypothèses et la formulation mathématique des problèmes sont posées :

- le calcul de probabilités est un problème de **quadrature**,
- le calcul du point du domaine de défaillance le plus probable (ou point de conception) est un problème d'**optimisation sous contraintes**,
- le classement des paramètres importants est traité à l'aide d'un **calcul de gradients**.

Nous exposons alors les méthodes de l'analyse de fiabilité et de sensibilité qui permettent de résoudre les problèmes précédents.

- **Méthodes d'approximation** : la fonction de modélisation étudiée est de type boîte noire, nous ne disposons pas du gradient.

  Notre choix se porte sur la construction d'une **approximation qui sera utilisée à la place du modèle initial**, nous présentons alors les méthodes d'approximation classiques.

  Nous détaillons les deux méthodes qui seront implémentées :

  - Les **réseaux de neurones** : un plan d'expérience de type **hypercube latin** permet d'obtenir les données d'apprentissage autour du point le plus probable et des techniques de **régularisation** sont employées.
  - L'interpolation par la technique des ***sparse grids*** : la sélection d'une partie seulement des fonctions de base (celles qui contribuent le plus à l'approximation) permet de travailler avec des grilles éparses qui réduisent fortement le coût de calcul normalement associé à ces méthodes.

    La construction de type **hiérarchique** et un **mode adaptatif** permettant de concentrer le coût de calcul dans les directions importantes en font un outil d'approximation puissant pour les problèmes en grande dimension.

- **Applications numériques** : nous présentons les cas tests fournis par le CEG et leur étude.

  Les cas tests sont définis par

  - une chaîne de logiciels modélisant le rejet d'un produit toxique, sa dispersion et les conséquences sur la population,
  - des scénarios, donnant le nombre (9 puis 30) de paramètres incertains, leurs intervalles de variation, et les lois de probabilité associées.

  Nous présentons la méthode d'approximation globale retenue pour l'étude :

  - calcul de probabilités avec une méthode de **Monte-Carlo**,
  - calcul du point de conception avec une **méthode classique** de descente,
  - classement des paramètres avec une **représentation de la distribution des gradients** selon tirage de type **carré latin**.

  Nous comparons alors les résultats obtenus avec les réseaux de neurones et les *sparse grids* sur le cas test à 9 paramètres.

La technique des *sparse grids* présentant de meilleures perspectives relativement au coût de calcul, nous la retenons pour le cas test à 30 paramètres : nous proposons alors des améliorations de la méthode globale qui s'avère insuffisante en grande dimension.

En ce qui concerne le calcul de probabilités, deux méthodes sont testées :

- Un **déplacement de la grille avec changement de variable** est effectué afin de **placer les points** de grille dans les **zones d'intérêt**, et d'améliorer l'approximation.
- Pour un seuil fixé, le **seuillage des données** permet d'obtenir une fonction plus facile à approcher.

Pour trouver le point de conception, une méthode par **approximations successives** est proposée.

Chapitre IV

Calculs de fiabilité

Sommaire

1 Traitement des incertitudes

1.1 Introduction

Actuellement pour des raisons d'économie de temps et de moyens, l'étude de systèmes physiques passe le plus souvent par une modélisation mathématique, puis par la simulation numérique. On réduit ainsi la durée du cycle de conception ou le nombre d'expériences à réaliser et donc les coûts de développement d'un produit.

Les méthodes numériques développées ces dernières années dans de nombreux domaines (mécanique des structures, mécanique des fluides, électromagnétisme,...) apportent une aide précieuse pour la compréhension des systèmes complexes modélisés.

Ces modèles servent souvent de prototype virtuel afin d'obtenir les meilleures caractéristiques du système. Pour les utiliser comme outil de *design* optimal, on définit des fonctions objectif à minimiser (poids, coût économique, ...) qui évaluent les performances du système. Certaines de ces performances peuvent avoir des valeurs limites ou critiques à ne pas dépasser (température, vibrations, ...).

L'utilisateur veut connaître généralement le meilleur *design*, mais aussi l'interprétation des résultats :

- il a besoin de savoir si la solution trouvée est robuste, et quels sont les paramètres les plus influents sur la réponse,
- il veut savoir dans quelle mesure il peut faire confiance à la réponse du modèle, à cause des incertitudes liées au modèle et aux calculs.

Les méthodes d'analyse de sensibilité et d'analyse d'incertitude ont été développées pour répondre à ces attentes et développer des modèles fiables : l'enjeu pour les ingénieurs est de pouvoir s'appuyer plus sûrement sur les modèles numériques dans leurs recherches.

De plus, au fur et à mesure que la puissance de calcul permet de développer des modèles de plus en plus complexes et proches de la réalité, le nombre de paramètres d'entrée devient très grand : l'utilisateur se trouve alors confronté à la limite du coût de calcul en grande dimension, dont les méthodes doivent tenir compte pour être utilisables en pratique.

Nous détaillons dans la suite plus précisément les différentes sources d'incertitude qui peuvent intervenir, en s'appuyant sur [WEF+01] et [Isu99].

1.2 Sources d'incertitudes

Un **modèle** est ici l'ensemble des équations gouvernant l'état d'un système et l'algorithme permettant de calculer le résultat.

Nous considérons pour notre étude une **modélisation numérique** : on fixe les valeurs numériques d'un certain nombre de paramètres d'entrée, et on obtient un résultat, nommé aussi **réponse** ou **sortie** du modèle, grâce à un code de calcul.

Incertitude naturelle Certains systèmes physiques ou biologiques sont considérés comme stochastiques (séismes, inondations,...), et d'autres comportent de façon inhérente des aspects stochastiques : ces caractéristiques aléatoires doivent alors être intégrées dans la modélisation.

On appelle cette incertitude naturelle ou aléatoire.

Certains paramètres sont alors considérés par définition aléatoires (la turbulence de l'atmosphère par exemple), mais d'autres, bien que mesurables précisément, sont aussi modélisés comme des paramètres aléatoires pour des raisons pratiques (pour limiter le coût lié à leur évaluation précise par exemple, comme pour les émissions d'une source dans l'atmosphère).

Incertitude décisionnelle Notre connaissance d'un système physique est nécessairement limitée, ainsi que l'information que l'on peut en retirer.

Les modèles mathématiques sont ainsi nécessairement une représentation simplifiée du phénomène étudié, et la construction du modèle repose sur le choix judicieux des hypothèses : un bon modèle apporte un compromis entre la simplification et la correcte évaluation des résultats du phénomène.

Ainsi, le choix du modèle apporte une incertitude dite décisionnelle, avec les aspects suivants, difficiles à appréhender dans l'analyse d'incertitude :

- Les hypothèses peuvent être simplifiées ou idéalisées : des paramètres sont parfois négligés pour des raisons d'efficacité (paramètres négligeables), ou par manque d'information (phénomène peu étudié, trop complexe,...). Le modèle ne décrit pas alors le comportement véritable du système.
- Le modèle peut être basé sur des ajustements empiriques à des expériences ou à des comportements observés, mais l'utilisateur peut hésiter entre plusieurs modèles concurrents.

Pour un phénomène donné, si plusieurs hypothèses entraînant des modèles différents sont susceptibles de le représenter, il faut s'assurer qu'ils donnent des résultats similaires, sinon il faut préciser davantage les hypothèses afin de trouver une modélisation robuste.

Incertitude du modèle Une fois le modèle choisi, ce dernier n'est valide que dans une région de l'espace des paramètres d'entrée définie par des frontières (limites en espace et en temps), et ses résultats dépendent de la résolution choisie pour chacune des discrétisations : une grille trop grossière introduit des résultats erronés.

On recherche ici aussi un compromis entre coût de calcul lié au pas de discrétisation et précision du résultat (dans certains cas, baisser la résolution n'entraîne pas une amélioration de la précision).

Les algorithmes de calcul entraînent également des approximations (arrondis et propagation d'erreurs) sur le résultat final.

L'incertitude de modèle est souvent évaluée en comparant la réponse du modèle à des résultats d'expériences, mais cette approche comporte des limitations, car les expériences sont elles mêmes entâchées d'incertitude (conditions aux limites, erreurs de mesure, ...).

Incertitude des paramètres d'entrée Les valeurs des paramètres d'entrée du modèle, ou celles de certaines constantes, sont établies généralement par mesure ou calibrage avec des données expérimentales. Ces données sont soit issues du tirage d'un échantillon puis évaluation, soit issues d'expériences non choisies.

Quand l'utilisateur doit fixer ces valeurs, il se trouve confronté en pratique à des difficultés pour les déterminer avec précision :

- erreurs de mesure, liées aux appareils ou aux conditions de mesure,
- échantillon non représentatif à cause de limitations en coût, non valide,...

- manque d'information : les outils employés pour prévoir la valeur recherchée n'incorporent pas tous les mécanismes qui expliquent la variabilité des paramètres.

Ainsi, l'incertitude des paramètres d'entrée provient du fait que même si on peut les mesurer théoriquement avec la précision voulue, et à fortiori pour les autres, leur codage numérique entraîne des erreurs : celles-ci se propagent lors des évaluations du modèle et peuvent générer des erreurs importantes sur le résultat.

Conclusion Les grandes catégories d'incertitudes et leurs recoupements sont illustrés par le schéma de la figure IV.1.

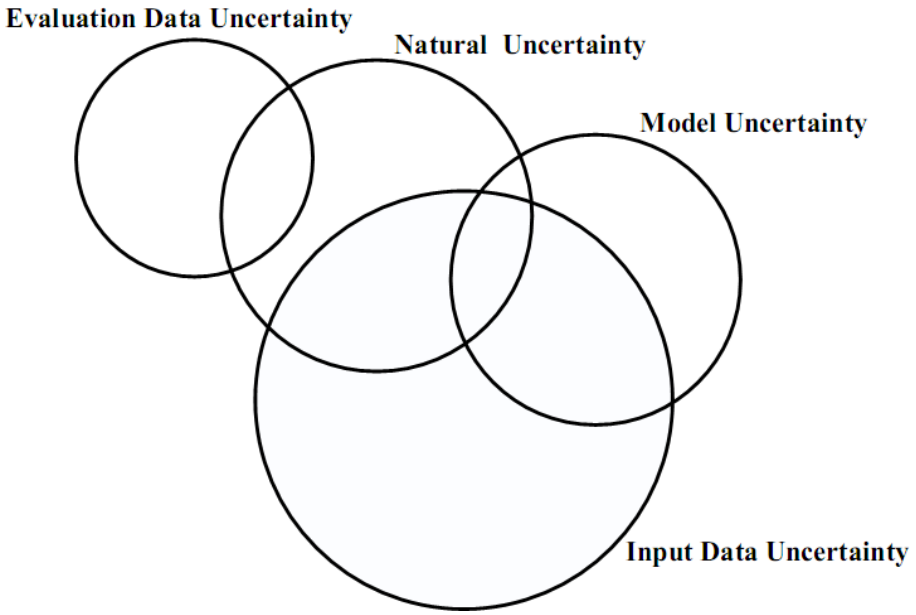


FIG. IV.1 – Catégories d'incertitude lors d'une modélisation (figure extraite de [Isu99])

L'incertitude du modèle et des paramètres d'entrée, nommée aussi incertitude épistémique, comporte les deux aspects suivants :

- L'**incertitude** dite **irréductible** : les fluctuations du résultat sont intrinsèques au problème étudié (par exemple la turbulence).

  Elle provient souvent de l'incertitude naturelle qui est encore présente, et ne peut être réduite que quand de nouvelles théories prenant en compte le phénomène de manière plus fine voient le jour.

- L'**incertitude** dite **réductible** : elle provient du fait que l'information concernant le système est toujours incomplète (faible nombre d'évaluations quand le modèle est coûteux, manque d'information sur une partie du système physique,...), et que le modèle, en tant qu'approximation, ne décrit pas exactement la réalité.

  Elle peut être réduite, dans une certaine mesure, en agissant aux différentes étapes du processus de modélisation :

- au niveau du modèle avec
  - des équations ou des hypothèses plus précises,
  - des moyens de calcul plus performants permettant de pousser la résolution,
  - des schémas numériques plus efficaces, ...
- au niveau des paramètres d'entrée avec
  - l'amélioration des mesures,
  - le recours à l'assimilation des données,...

Mais dans certains cas, il est plus judicieux de prendre en compte une incertitude réductible sous forme de bruit que d'essayer de la réduire au prix d'une augmentation du coût de calcul excessive.

L'exploitation des résultats de modèles complexes rattachés à des systèmes physiques doit donc de préférence s'accompagner d'une évaluation des incertitudes associées : tous les aspects sont importants, mais certains sont plus difficile à prendre en compte (incertitude naturelle et incertitude du modèle). L'incertitude sur le résultat provient ainsi des choix effectués à chaque étape de la modélisation et du calcul numérique.

Nous nous intéressons à la prise en compte de l'incertitude irréductible, en observant les points suivants :

- caractérisation de l'incertitude des paramètres d'entrée du modèle définis comme des variables aléatoires,
- **étude de la propagation de l'incertitude** au cours des calculs : le résultat est une variable aléatoire dont on calcule des mesures de probabilité comme l'espérance, la variance, ou la fonction de répartition,...
- validation par l'étude du système par comparaison des résultats de simulation avec des expériences réelles.

Nous nous limitons dans cette étude au deuxième point, en prenant comme hypothèses pour la suite :

- un **modèle** de système physique, de type "**boîte noire**" (nous n'avons pas accès aux équations), est donné sous forme de **code numérique**,
- les **lois de probabilité des paramètres d'entrée sont données**,
- la **sortie** ou réponse du modèle est un critère qui décrit le système, et son évaluation est considérée comme **très coûteuse** car le **nombre de paramètres est grand**.

Remarque : couplage modèle-données Dans le cadre de l'incertitude irréductible, les erreurs non modélisables peuvent être prises en compte en couplant le modèle avec les données, ou mesures d'entrée.

Cette approche est notamment utilisée dans la modélisation météorologique, et dans le domaine plus large de l'assimilation de données.

1.3 Approches possibles

1.3.1 Introduction

Les différentes approches dépendent des hypothèses effectuées sur la modélisation de l'incertitude :

- L'**approche probabiliste** est la plus répandue : elle consiste à modéliser les incertitudes sur les paramètres d'entrée par des variables aléatoires.

  L'objectif est alors d'**estimer les caractéristiques de variabilité de la réponse** (espérance, variance,...) qui est aussi une variable aléatoire.

  Elle nécessite néanmoins de disposer d'information sur les paramètres d'entrée afin de choisir la densité de probabilité la mieux adaptée à leurs variations.

  Nous détaillons cette approche en section 1.3.2, car c'est celle que nous utiliserons dans le cadre de notre étude.

- Quand on ne dispose pas d'information suffisante, on se donne uniquement des bornes sur les variations des paramètres d'entrée, et on cherche à établir des bornes sur la réponse du modèle : c'est le domaine de l'**analyse de robustesse**.

  On utilise alors des calculs d'arithmétique par intervalles (voir par exemple [Moo79]) pour étudier la propagation des incertitudes jusqu'au résultat.

  Cette approche possède un défaut majeur : elle peut être très pessimiste.

- L'introduction de la **logique floue** (ou *fuzzy logic*) permet l'incorporation de concepts différents issus de la théorie de l'information.

  On étudie la propagation d'ensembles flous modélisant l'incertitude avec de nouveaux paramètres d'entrée comme le jugement subjectif et l'opinion experte (exclus dans l'approche probabiliste).

1.3.2 Approche probabiliste

Cette approche est la plus habituelle car elle repose sur la théorie des probabilités, développée depuis longtemps, et adaptée à la modélisation des effets du hasard et donc aux incertitudes.

On se donne un **modèle probabiliste : chaque paramètre d'entrée incertain est modélisé par une variable aléatoire** (voir [DM96]) sous forme de la densité de probabilité la plus adéquate.

Choix de la densité de probabilité Choisir la densité de probabilité adaptée est une étape importante de la modélisation stochastique, car les résultats de l'analyse qui s'en suit dépendent fortement de la qualité des données.

En pratique, la densité de probabilité d'un paramètre est estimée :

- A l'aide des fréquences observées avec des données déjà disponibles ou en générant un échantillon représentatif : on peut estimer les moments statistiques (moyenne, variance).

  On peut ensuite ajuster une densité de probabilité à l'aide d'un test statistique ou grâce à des familles de densités déjà évaluées.

- Ou à partir de l'expérience de l'ingénieur dans la modélisation du système étudié.

La loi normale est souvent utilisée grâce à ses bonnes propriétés mathématiques (additivité et théorème central limite).

Etude des incertitudes Ensuite, on étudie la propagation de l'incertitude par l'analyse de **la réponse du modèle** qui est **aussi une variable aléatoire**.

En mécanique par exemple, cette approche a été féconde, et donné lieu à une branche appelée mécanique stochastique (voir [Sch01] ou [DKP86]).

On peut distinguer les objectifs complémentaires suivants :

- L'**analyse de sensibilité** (ou analyse des moments) cherche à estimer la densité de probabilité du résultat (par exemple en calculant sa moyenne et sa variance si on considère que la réponse suit une loi normale).

  On peut aussi établir des intervalles de confiance sur les paramètres d'entrée et on obtient alors un intervalle de confiance sur la réponse.

  Par exemple dans [Sch06], des quantiles de la réponse, qui est une production de pétrole, sont calculés pour prendre des décisions dans un contexte incertain.

  En mécanique, cette analyse est en général conduite autour du point moyen de fonctionnement, et permet de déterminer si la variabilité d'un paramètre est amortie ou au contraire amplifiée par le modèle : dans ce dernier cas apparaissent les risques d'instabilité.

- L'**analyse de fiabilité** cherche à étudier les conditions et les risques de défaillance du système modélisé.

Dans la suite nous nous plaçons dans le cadre de l'approche probabiliste.

1.3.3 Analyse de sensibilité

Le lecteur trouvera un exposé complet sur les méthodes suivantes dans [STCM04] et [SCS00].

Définition Nous considérons

- les **variations du modèle**, qui correspondent aux variations de la fonction associant les entrées à la sortie,
- la **variabilité du modèle**, qui correspond dans le cadre de l'approche probabiliste aux mesures de dispersion comme la variance de la sortie.

Dans la suite, sauf précision, nous utiliserons le terme de "variations" au sens large, en sous-entendant aussi la variabilité.

L'**analyse de sensibilité** consiste à effectuer l'**évaluation, qualitative ou quantitative, des variations ou de la variabilité de la réponse du modèle à partir de la variation ou de la variabilité des paramètres d'entrée**.

Objectifs Les objectifs dépendent de l'utilisateur du modèle :

- Pour le spécialiste qui veut explorer un système ou processus coûteux en définissant un modèle numérique, l'analyse de sensibilité apporte une évaluation de la confiance que l'on peut accorder au modèle.

  On peut calculer l'incertitude propagée au résultat à partir des erreurs ou incertitudes estimées sur les paramètres.

- Pour l'ingénieur, l'objectif est souvent de trouver un compromis entre comportement du système, qu'on veut optimal (par exemple la production maximale d'un puits comme dans [Sch06]) et le niveau d'incertitude sur le résultat, qu'on veut minimal. On veut déterminer les paramètres influents, afin d'optimiser le système étudié.

  Par exemple dans le domaine du *design* elle sert à identifier les paramètres sur lesquels agir pour trouver la forme optimale.

- Pour un chimiste, l'analyse de sensibilité permet de mesurer la force de la relation entre les entrées et les sorties d'une réaction.

- Pour un économiste, elle permet de quantifier la stabilité des coefficients d'un modèle de régression (analyse de robustesse) ...

Le principe commun est d'étudier l'effet des paramètres d'entrée :

- déterminer les paramètres influents, ceux dont une faible variation engendre une grande variation de la réponse, et dont l'étude est la plus importante,

- déterminer les paramètres négligeables, ceux qui n'affectent pas les variations de la réponse,

- identifier si une région de l'espace des paramètres correspond aux variations maximales de la réponse,

- éventuellement déceler comment les paramètres interagissent entre eux.

L'utilisateur peut alors quantifier les incertitudes sur les paramètres les plus influents : si les risques sont importants, il peut améliorer leur traitement par une plus grande précision (nouvelles mesures pour affiner les valeurs des paramètres déficients).

En ce qui concerne une étude en grande dimension, la possibilité d'identifier les paramètres négligeables est primordiale, pour les éliminer du modèle final. On peut ainsi réduire la dimension du problème, car la prise en compte de variables non significatives augmente la complexité et le coût de calcul sans forcément améliorer les conclusions.

Nous détaillons certaines méthodes d'analyse de sensibilité dans la section 5.1.

1.3.4 Etude de fiabilité

Comme expliqué précédemment :

- la modélisation ne peut être parfaite, et il paraît raisonnable de dire qu'il existera toujours un écart non modélisable avec la réalité physique,
- de plus, le hasard intervient dans le processus, et il faut le prendre en compte,
- de petites perturbations de l'état initial peuvent conduire à des solutions très différentes, et il faut essayer de maîtriser ces phénomènes d'instabilité.

L'objet de la fiabilité est de s'assurer que la maîtrise de l'incertitude est suffisante pour que les risques soient bien évalués et restent acceptables.

La définition de la fiabilité donnée par l'AFNOR (pour Association Française de Normalisation, qui édite les normes NF) est : "Aptitude d'un dispositif à accomplir une fonction requise dans des conditions données, pendant une durée donnée... le terme est aussi utilisé comme caractéristique désignant une probabilité de succès ou un pourcentage de succès".

La définition précédente oriente l'étude vers l'aproche probabiliste, qui n'est pas la seule, mais qui s'est ainsi fortement développée, accompagnée par les connaissances en probabilités et en statistiques, et s'est étendue à d'autres domaines (conception des systèmes électroniques par exemple).

Définition Une **étude de fiabilité** consiste à analyser les conditions de défaillance du système ou processus modélisé afin d'apporter une aide à la décision ou à la conception : on calcule les **probabilités des états de défaillance** et le **point de défaillance le plus probable**.

Mécanique des structures On trouve des applications du calcul de fiabilité notamment dans le domaine du calcul de structures. Les techniques de calcul de fiabilité dans le domaine de la mécanique sont par exemple présentées dans [Lem05].

En effet, les constructions d'ouvrages ou de machines nouvelles se sont toujours accompagnées du souci d'intégrer le risque de la destruction d'une partie, aux conséquences potentiellement catastrophiques.

L'ingénieur, dans un souci de conception optimale, doit s'assurer de construire au meilleur coût un système : l'évaluation des coûts ou des conséquences d'une panne ou d'une destruction, c'est-à-dire une mesure de la **gravité** de la défaillance, associées au calcul d'une **probabilité de défaillance**, permettent de déterminer le **risque** encouru avec la relation suivante (voir [Lem05])

$$\text{Risque} = \text{Probabilité} \times \text{Gravité}$$

Les domaines d'applications industrielles sont nombreux : la fiabilité des structures marines *offshore*, l'identification des gisements de pétrole ou de minerais par étude d'incertitudes (voir [Sch06] et section 1.5.3 page 199 du chapitre suivant), les normes

de fiabilité des constructions, la maintenance par contrôle des défaillances des stuctures (pièces de moteur,...).

Avec la progression de la connaissance des lois mécaniques, le travail des ingénieurs a permis des modélisations de plus en plus proches de la réalité. Les outils de calcul en mécanique des structures ont ainsi atteint rapidement une grande maturité, ce qui a poussé les chercheurs à explorer deux nouvelles directions :

- la conception,
- le calcul des incertitudes.

2 Introduction du problème

2.1 Données

Nous allons donner les notations qui forment le cadre de notre étude, puis présenter les méthodes de calculs de fiabilité, en nous appuyant sur [Yar03], [Lem05], [MKL86].

2.1.1 Fonction de modélisation

Nous étudions une **fonction de modélisation** notée G du type suivant :

$$\begin{array}{rccc} G: & \Omega \subset \mathbb{R}^n & \longrightarrow & \mathbb{R} \\ & X = (x_1, \ldots, x_n) & \longmapsto & G(X). \end{array} \tag{IV.1}$$

Nous cherchons à obtenir des informations sur les incertitudes liées au phénomène modélisé par G en étudiant le comportement de cette fonction : G représente un code de calcul.

Nous nous plaçons donc dans le cadre d'**expériences déterministes simulées** :

- les variables x_i sont appelées **paramètres d'entrée** : leurs valeurs varient dans un intervalle $[m_i, M_i]$ et le domaine d'étude est le pavé $\Omega = \prod_{i=1}^{n} [m_i, M_i]$.
- la **réponse**, ou **sortie** $G(X)$ est calculée pour un point X quand les valeurs des paramètres sont fixées.

Nous considérons alors les hypothèses suivantes, qui guideront le choix de la méthode :

- Nous travaillons dans le contexte de la **grande dimension** : le nombre de paramètres n est de l'ordre de plusieurs dizaines.
- La fonction G n'est connue que par des simulations, nous n'avons pas accès aux équations du modèle (modèle type boîte noire). Le gradient de G n'est pas disponible (sauf si on calcule une approximation).
- Chaque simulation est coûteuse en temps (c'est le cas de la plupart des codes industriels).

Ces hypothèses correspondent à la problématique rencontrée dans de nombreux domaines de la simulation numérique : code d'éléments finis pour évaluer les performances d'une structure, code de la dynamique des fluides pour un calcul aérodynamique,... En mécanique des structures par exemple, le critère qui définit G peut être une mesure des déplacements, des déformations, des contraintes,...

Si le modèle possède plusieurs sorties (voir figure IV.2), il faut faire une étude différente pour chaque sortie définissant une fonction G du type ci-dessus.

Remarque Si l'utilisateur n'a pas à sa disposition un code de modélisation, il ne peut effectuer que des expériences réelles non reproductibles et pas toujours réalisables ou ne dispose que d'un tableau de données issues d'expériences non choisies. On ne peut pas alors définir de fonction de modélisation au sens ci-dessus, qui implique l'utilisation de méthodes différentes.

2.1.2 Lois de probabilité des paramètres

Nous supposons que l'incertitude sur la valeur des paramètres d'entrée est représentée par des lois de probabilité données au départ (voir section 1.3.2 et pour plus de précisions, voir par exemple [Gar83]).

Nous présenterons les paramètres utilisés dans notre étude dans la section des résultats numériques. En mécanique des structures par exemple, ces paramètres sont la géométrie du système, les propriétés des matériaux, le chargement,...

Notations Nous nous plaçons dans la suite dans le cadre suivant :

- $\mathbf{X} = (\mathbf{x_1}, \ldots, \mathbf{x_n})$ est un **vecteur aléatoire** de $\mathbb{R}^n$ définissant les paramètres d'entrée,
- $\rho(\mathbf{X})$ est la **densité de probabilité** jointe de $\mathbf{X}$, avec
  - $\rho(\mathbf{X}) = \rho(\mathbf{x_1}) \ldots \rho(\mathbf{x_n})$ car les variables aléatoires $\mathbf{x_j}$, $j = 1 \ldots n$ sont supposées **indépendantes**,
  - comme les paramètres sont définis sur des intervalles, les lois à support dans $\mathbb{R}$ sont tronquées,
  - $p(\mathbf{X} \in D) = \int_D \rho(\mathbf{X}) d\mathbf{X}$ la probabilité d'appartenir au domaine $D \subset \mathbb{R}^n$.

Le point le plus probable de l'espace des paramètres est appelé **point nominal**.

2.1.3 Loi de probabilité de la réponse

On peut calculer, pour une réalisation $X \in \mathbb{R}^n$ du vecteur aléatoire $\mathbf{X}$, la valeur $G(X)$: on obtient ainsi une **variable aléatoire** $G(\mathbf{X})$: $\Omega \subset \mathbb{R}^n \longrightarrow \mathbb{R}$, qui représente l'incertitude sur le résultat.

L'étude des caractéristiques de la densité de probabilité de $G(\mathbf{X})$ nous donne l'incertitude de la réponse du modèle (voir pour illustration la figure IV.2).

Dans la suite, nous notons $F_G(s) = p(G(\mathbf{X}) < s)$ la **fonction de répartition** associée à $G(\mathbf{X})$.

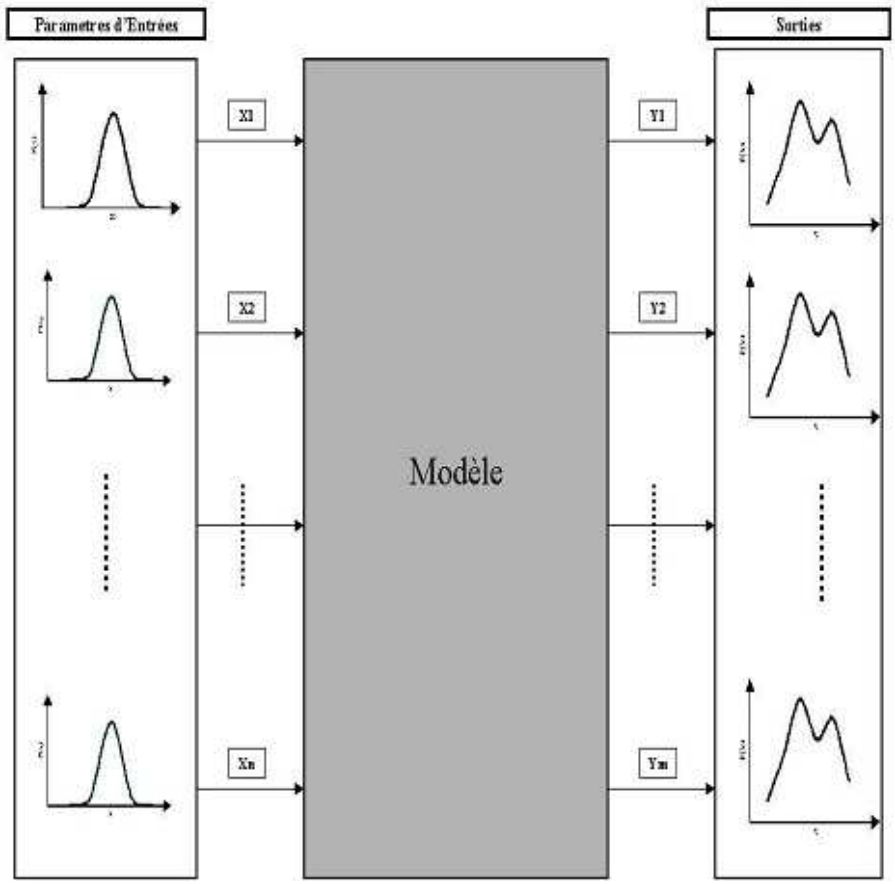


FIG. IV.2 – Propagation des incertitudes dans un modèle (image extraite de [Isu99])

2.1.4 Seuil de défaillance s

On définit $s \in \mathbb{R}$ un **seuil de défaillance**, qui correspond en pratique au seuil d'une grandeur physique (tension, puissance totale reçue,...) au-delà duquel on observe une défaillance, ou un dysfonctionnement du système modélisé par G.

2.1.5 Domaine de défaillance D_s

On définit le **domaine de défaillance** noté D_s associé au seuil s par

$$D_s = \{X \in \mathbb{R}^n;\ s \leqslant G(X).\} \tag{IV.2}$$

Le domaine de défaillance correspond aux valeurs des paramètres d'entrée dont l'image par G est supérieure au seuil de défaillance : en pratique, il définit une zone critique.

Remarque En mécanique des structures par exemple, on définit $G : \mathbb{R}^n \longrightarrow \mathbb{R}$ de façon à ce que le domaine de défaillance soit défini par les valeurs négatives de G, c'est-à-dire :

- $D = \{X \in \mathbb{R}^n;\ G(X) \leqslant 0\}$ est le domaine de défaillance,
- $\{X \in \mathbb{R}^n;\ G(X) = 0\}$ est la surface limite,
- $\{X \in \mathbb{R}^n;\ G(X) > 0\}$ le domaine de sûreté (il correspond aux valeurs des paramètres qui assurent un comportement normal de la structure).

Nous conservons la notation associée à s car nous voulons obtenir des résultats pour un seuil s variable.

2.1.6 Transformation isoprobabiliste

Espace gaussien standard Un vecteur aléatoire $\mathbf{U}$ appartient à l'espace dit gaussien standard si ses composantes $\mathbf{u_j}$:

- suivent toutes une loi gaussienne centrée réduite (écart-type unitaire),
- sont indépendantes deux à deux.

Dans cet espace représenté en dimension deux, les courbes d'iso-probabilité sont des cercles centrés à l'origine O, qui est le point le plus probable.

Transformation T La **transformation isoprobabiliste** T (voir figure IV.3 pour une illustration) permet de passer d'un espace de lois de probabilités quelconques, dit espace physique, à l'espace gaussien standard :

$$\begin{array}{rccc} T: & \mathbb{R}^n & \longrightarrow & \mathbb{R}^n \\ & \mathbf{X} = (\mathbf{x_1}, \ldots, \mathbf{x_n}) & \longmapsto & T(\mathbf{X}) = \mathbf{U} = (\mathbf{u_1}, \ldots, \mathbf{u_n}). \end{array} \tag{IV.3}$$

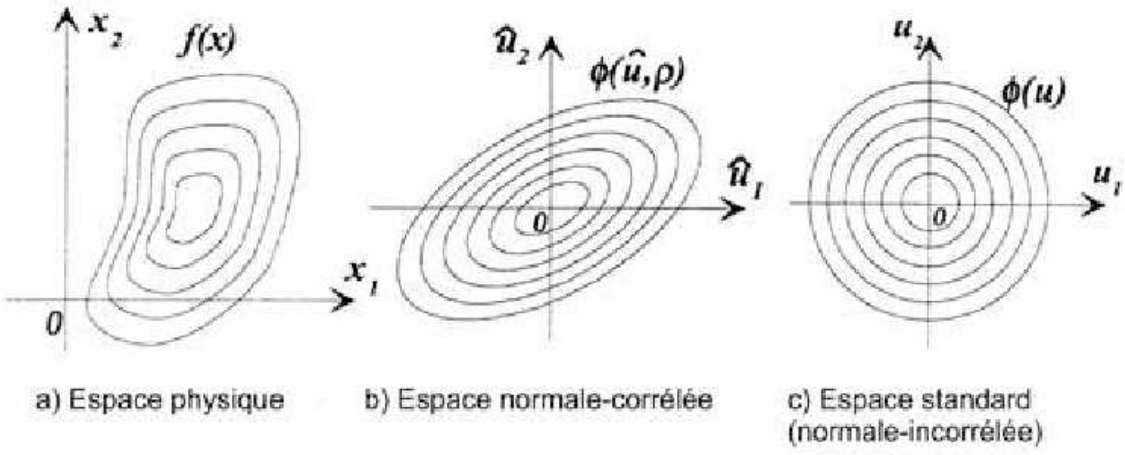


FIG. IV.3 – Illustration de la transformation de Rosenblatt pour deux paramètres

Remarques

- Quand les variables de l'espace physique $\mathbf{x_j}$ suivent une loi gaussienne d'espérance μ_j et d'écart type σ_j et sont indépendantes, T désigne l'opération affine de réduction et centrage

$$\begin{array}{ccc} \mathbb{R} & \longrightarrow & \mathbb{R} \\ \mathbf{x_j} & \longmapsto & \frac{\mathbf{x_j}-\mu_j}{\sigma_j}. \end{array} \tag{IV.4}$$

- La densité de probabilité d'une loi normale centrée réduite est

$$\begin{array}{rccl} \rho: & \mathbb{R} & \longrightarrow & [0,1] \\ & \mathbf{x_j} & \longmapsto & \rho(\mathbf{x_j}) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x_j^2}{2}}. \end{array} \tag{IV.5}$$

- Suivant la quantité d'information disponible à propos des variables $\mathbf{x_j}$, plusieurs transformations sont possibles.

  La transformation la plus connue est celle de Rosenblatt (introduite dans [Ros52], voir aussi [HL74]) :

  - elle demande de connaître la densité de probabilité jointe $\rho(\mathbf{X})$ (ce qui est parfois difficile en pratique),
  - elle a une écriture unique et simplifiée dans le cas de variables aléatoires $\mathbf{x_j}$ indépendantes deux à deux,
  - elle n'est en général pas connue analytiquement (il faut l'évaluer numériquement) et pas unique.

  En général, on ne dispose que des densités de probabilités $\rho(\mathbf{x_j})$ marginales et des coefficients de corrélation, qui ne permettent pas de déduire $\rho(\mathbf{X})$.

  Des techniques approchées comme la transformation de Nataf (voir [Nat62] et [DKP86]) permettent alors d'effectuer le changement de variables.

2.2 Problèmes à résoudre

Nous nous plaçons dans le cadre des hypothèses précédentes. Nous considérons une fonction de modélisation G et un seuil s donnés, et le domaine de défaillance D_s associé.

Nous nous donnons comme objectif de résoudre les problèmes suivants.

2.2.1 (P_1) Probabilité de défaillance p_s

Le but du calcul de fiabilité est de calculer la probabilité des états de défaillance, c'est-à-dire d'appartenir au domaine de défaillance, ou encore la **probabilité de dépasser un seuil** s, définie par

$$(P_1) \quad p_s := p(s \leqslant G(\mathbf{X})) = p(\mathbf{X} \in D_s). \tag{IV.6}$$

En introduisant la densité de probabilité $\rho(\mathbf{X})$, cette probabilité se ramène à un problème de calcul d'intégrale multiple en grande dimension

$$p_s = \int_{D_s} \rho(\mathbf{X}) d\mathbf{X} = \int_{D_s} \rho(\mathbf{x_1}) \dots \rho(\mathbf{x_n}) d\mathbf{x_1} \dots d\mathbf{x_n}. \tag{IV.7}$$

Caluler p_s revient à estimer la fonction de répartition F_G, et donc $\rho(\mathbf{X})$, car $p_s = 1 - F_G(s)$.

Espérance Dans les méthodes classiques d'analyse d'incertitude, on estime la loi de probabilité de la réponse $G(\mathbf{X})$ en donnant une estimation de sa densité de probabilité ou les principaux moments associés (espérance, variance,...).

Le calcul de l'espérance de $G(\mathbf{X})$ se ramène aussi au calcul d'une intégrale avec

$$E[G(\mathbf{X})] = \int G(\mathbf{X})\ \rho(\mathbf{X}) d\mathbf{X}. \tag{IV.8}$$

A l'inverse, le calcul de la probabilité de défaillance s'écrit comme un calcul d'espérance car

$$\int_{D_s} \rho(\mathbf{X}) d\mathbf{X} = \int_{\mathbb{R}^n} \mathbb{I}_{D_s}(\mathbf{X}) \rho(\mathbf{X}) d\mathbf{X} \tag{IV.9}$$

donc

$$p_s = E[\mathbb{I}_{D_s}(\mathbf{X})] \tag{IV.10}$$

avec $\mathbb{I}_{D_s}(X) = \begin{cases} 1 & \text{si } X \in D_s \Leftrightarrow s \leqslant G(X) \\ 0 & \text{si } X \notin D_s \Leftrightarrow s > G(X) \end{cases}$ la fonction indicatrice sur D_s.

2.2.2 (P_2) Points de conception (MPP)

Le ou les **points de conception**, sont définis comme les **points P^* du domaine de défaillance les plus probables** (appelés aussi MPP pour *Most Probable Point*).

Problème d'optimisation Nous travaillons dans l'espace gaussien standard (voir section 2.1.6 pour la transformation T) : comme le point le plus probable est l'origine O, le point de conception est le point du domaine de défaillance dont la distance à O est minimum.

Cela amène la formulation du problème d'optimisation sous contrainte suivant

$$(P_2) \quad P^* = \min_{U \in \overline{D_s}} \|\mathbf{U}\|_2^2 = \min_{U \in \overline{D_s}} U^T U \tag{IV.11}$$

avec le domaine de défaillance dans l'espace gaussien standard
$\overline{D_s} := \{U \in \mathbb{R}^n;\ s \leqslant H(U)\}$ en posant $H := G \circ T^{-1}$.

Pour avoir les coordonnées de P^* dans l'espace physique, on calcule $T^{-1}(P^*)$.

Une illustration de point de conception avec deux variables aléatoires $\mathbf{x_1}$ et $\mathbf{x_2}$ dans l'espace gaussien standard est présentée dans la figure IV.5 page 166.

Remarque La dénomination de point de conception vient de la mécanique, où G modélise le critère de conception d'une structure.

Dans notre étude, les points de conception correspondent aux valeurs de paramètres que nous voulons éviter (car donnant un résultat critique), les plus susceptibles d'apparaître.

2.2.3 (P_3) Paramètres importants

L'expérience des études de phénomènes dépendant d'un grand nombre de paramètres montre que la plupart du temps, seul un petit nombre parmi ceux-ci sont significatifs.

Un **paramètre** x_i est **important**, ou **influent**, si ses **variations ont une influence prépondérante sur la variation du résultat** $G(X)$.

Nous voulons un critère qui permet de savoir quels paramètres sont importants, et si d'autres au contraire sont **négligeables** : leurs variations n'influencent pas la réponse.

D'un autre point de vue, on cherche à connaître la part d'incertitude de chaque paramètre x_i sur l'incertitude de la réponse $G(X)$ du modèle. On peut répondre à cette question grâce à l'analyse de sensibilité, présentée en section 5.1 page 172.

Utilité : réduction de modèle Cela donne une information importante sur le modèle étudié : il faudra alors surveiller en priorité la variation des paramètres importants (par exemple par des contrôles de qualité en mécanique), ce qui permet de mieux maîtriser son comportement.

L'analyse de sensibilité nous informe sur les paramètres dont l'influence est amplifiée ou au contraire atténuée.

Cette approche permet ensuite de réduire la taille du problème, car on peut alors restreindre l'étude de fiabilité aux paramètres importants, considérés comme les seuls véritablement source d'incertitude, et négliger ou faire une approximation sur les paramètres restants.

C'est une façon de répondre à la difficulté du coût de calcul en très grande dimension en simplifiant le problème.

2.2.4 Objectifs

Nous devons présenter une méthode efficace, car le but de l'étude est de livrer un logiciel d'aide à la décision, qui doit pouvoir être utilisé avec un temps de calcul modéré : cette contrainte est essentielle pour guider notre choix.

Les autres éléments à considérer sont :

- la méthode doit pouvoir fonctionner en grande dimension pour une implémentation ultérieure (une centaine de paramètres),
- elle doit permettre de résoudre les trois problèmes précédents.

3 Méthodes de calcul des probabilités de défaillance p_s

Nous voulons résoudre le problème (P_1) de quadrature en grande dimension (voir section 2.2.1).

Principe L'évaluation par intégration directe ne peut pas s'effectuer en pratique, sauf dans le cas de systèmes simples pour lesquels la fonction de modélisation est connue analytiquement.

Il faut alors utiliser une méthode d'évaluation numérique, dite de quadrature : dans la suite, nous présentons les différentes méthodes d'intégration numérique.

3.1 Formules de quadrature

Les formules de quadrature de Gauss permettent de calculer la valeur approchée d'une intégrale en dimension un. Elles se généralisent par produit tensoriel au calcul des intégrales multi-dimensionnelles : le résultat est une combinaison linéaire d'évaluations de la fonction en des points appelés noeuds.

Pour une quadrature en dimension d, avec m noeuds dans chaque direction, on doit évaluer la fonction sur une grille de m^d noeuds : ce nombre croît de façon exponentielle avec la dimension, et cette approche devient impraticable quand d est grand.

3.2 Méthodes de simulation

Ce sont des méthodes de tirage qui utilisent un échantillonnage du vecteur aléatoire $\mathbf{X}$: on utilise alors des critères statistiques sur les évaluations de l'échantillon par G.

3.2.1 Méthode de Monte-Carlo

Estimateurs Dans le cas du vecteur aléatoire $G(\mathbf{X})$ et de la probabilité de défaillance p_s :

- on effectue N tirages indépendants (ou N réalisations) $\{X_i,\ i = 1, \ldots, N\}$ suivant la loi de probabilité de $\mathbf{X}$,
- on leur applique la fonction de modèlisation et on obtient les réalisations $\{G(X_i),\ i = 1, \ldots, N\}$.

En appliquant la moyenne empirique, on obtient les **estimateurs** respectivement $\widehat{E_{GN}}$ de l'**espérance** de $G(\mathbf{X})$ et $\widehat{p}_N$ de la **probabilité de défaillance** p_s (voir formules (IV.9) et (IV.10)) :

$$\widehat{E_{GN}} := \frac{1}{N}\sum_{i=1}^{N} G(X_i) \quad \simeq \quad E[G(\mathbf{X})], \tag{IV.12}$$

$$\widehat{p}_N := \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_{D_s}(X_i) \quad \simeq \quad E[\mathbb{I}_{D_s}(\mathbf{X})] = p_s. \tag{IV.13}$$

Remarque En pratique, on effectue les tirages avec un code qui génère des nombres de manière déterministe : ces nombres sont dits pseudo-aléatoires.

Ces estimateurs sont sans biais, et la loi des grands nombres assure qu'ils convergent presque sûrement, c'est-à-dire respectivement pour $\widehat{p}_N$

$$E(\widehat{p}_N) = p_s \quad \text{et} \quad \lim_{N\to+\infty} \widehat{p}_N = p_s. \tag{IV.14}$$

Erreur Etant donné son caractère stochastique, la méthode de Monte-Carlo ne fournit qu'une estimation d'erreur probabiliste.

Quand $N \to +\infty$, la loi de l'estimateur $\widehat{p}_N$ devient une loi Gaussienne : l'erreur, ou précision, de la méthode est alors donnée par une estimation de l'écart-type de $\widehat{p}_N$ (voir [Fis96] ou [LPS98]).

On calcule la variance de $\widehat{p}_N$ avec

$$\begin{aligned} Var(\widehat{p}_N) &= Var\left[\frac{1}{N}\sum_{i=1}^{N} \mathbb{I}_{D_s}(X_i)\right] \\ &= \frac{1}{N^2}\, Var\left[\sum_{i=1}^{N} \mathbb{I}_{D_s}(X_i)\right] \\ &= \frac{1}{N^2}\, \sum_{i=1}^{N} Var\left[\mathbb{I}_{D_s}(X_i)\right] \\ &= \frac{1}{N^2} N\, Var\left[\mathbb{I}_{D_s}(X_i)\right] \\ &= \frac{1}{N}\, Var\left[\mathbb{I}_{D_s}(X_i)\right] \end{aligned}$$

car les réalisations de $\mathbb{I}_{D_s}(X_i)$ sont indépendantes et identiquement distribuées.

Or, comme $\mathbb{I}_{D_s}^2 = \mathbb{I}_{D_s}$ alors $E[\mathbb{I}_{D_s}(\mathbf{X})^2] = E[\mathbb{I}_{D_s}(\mathbf{X})]$, et le calcul de la variance devient

$$\begin{aligned} Var(\widehat{p}_N) &= \frac{1}{N}\, \left(E[\mathbb{I}_{D_s}(\mathbf{X})^2] - E[\mathbb{I}_{D_s}(\mathbf{X})]^2\right) \\ &= \frac{1}{N}\, \left(E[\mathbb{I}_{D_s}(\mathbf{X})](1 - E[\mathbb{I}_{D_s}(\mathbf{X})])\right) \\ &= \frac{1}{N}\, p_s(1-p_s). \end{aligned}$$

Ainsi l'erreur est de l'ordre de l'écart-type

$$\widehat{\sigma}_N := \sqrt{Var(\widehat{p}_N)} = \frac{1}{\sqrt{N}}\, \sqrt{p_s(1-p_s)}. \tag{IV.15}$$

On peut calculer alors une approximation de $\widehat{\sigma}_N$ en remplaçant p_s par $\widehat{p}_N$

$$\widehat{\sigma}_N \simeq \frac{1}{\sqrt{N}}\, \sqrt{\widehat{p}_N(1-\widehat{p}_N)}. \tag{IV.16}$$

Cela permet d'écrire une **estimation de l'intervalle de confiance** à 95% pour $\widehat{p}_N$ sous la forme

$$[\widehat{p}_N - 1.96\,\widehat{p}_N, \widehat{p}_N + 1.96\,\widehat{p}_N]\,. \tag{IV.17}$$

Utilisation pratique La méthode de Monte-Carlo présente les avantages suivants :

- la simplicité de mise en oeuvre : il suffit de générer des tirages aléatoires,
- la robustesse : elle s'applique à n'importe quel type de fonctions, même non régulières,
- l'erreur d'estimation est indépendante de la taille du problème, c'est-à-dire du nombre de paramètres d'entrée.

Cela explique l'usage très répandu dans de nombreux domaines de cette méthode pour résoudre des problèmes de quadrature numérique.

Mais elle présente également un inconvénient majeur : comme le montre la formule (IV.15), la vitesse de convergence est lente, de l'ordre de $\frac{1}{\sqrt{N}}$. La méthode requiert un nombre excessif d'appels à la fonction d'évaluation G pour obtenir une précision croissante de l'estimateur.

Par exemple, si on écrit le coefficient de variation sous la forme

$$c := \frac{\widehat{\sigma}_N}{p_s} = \frac{\sqrt{1-p_s}}{\sqrt{N p_s}} \simeq \frac{1}{\sqrt{N p_s}} \tag{IV.18}$$

quand p_s est petit.

Alors par exemple pour un coefficient de variation $c = 0.1$ et une probabilité $p_s = 10^{-n}$, le nombre d'évaluations est $N = 10^{n+2}$.

Dans le cas d'un problème en grande dimension, ou pour un modèle couteux à évaluer, cette méthode devient impraticable à cause du coût de calcul.

Généralement, quand le seuil est très grand, alors le domaine de défaillance est petit (ce qui est le cas quand on calcule une probabilité très faible d'un événement rare, comme la rupture d'une pièce) : la difficulté réside dans le fait d'accéder à l'information utile. Le générateur de nombres pseudo-aléatoires ne parvient pas à placer des tirages dans la zone de défaillance qui est peu probable, c'est pourquoi la méthode de Monte-Carlo présente alors des erreurs importantes.

3.2.2 Méthodes de quasi Monte-Carlo : réduction de variance

Des améliorations de la méthode de Monte-Carlo ont été apportées : on cherche à réduire la variance de l'estimateur, c'est-à-dire augmenter sa précision et accélérer la convergence, grâce à un meilleur échantillonnage que le tirage aléatoire simple.

L'idée est de réduire variance de l'estimateur en augmentant la densité des tirages dans les régions intéressantes : à cet effet, on privilégie les régions qui contribuent le plus à l'estimation, c'est-à-dire qui apportent le plus d'information.

En effet, le caractère aléatoire des nombres générés a moins d'importance que leur distribution uniforme. L'idée dans les méthodes de quasi Monte-Carlo est alors de

remplacer les suites de nombres aléatoires par des suites de nombres déterministes avec une meilleure distribution : ces suites sont dites quasi-aléatoires ou à faible discrépance.

Le comportement asymptotique est meilleur que pour Monte-Carlo : on réduit le nombre de tirages nécessaires pour atteindre une précision donnée.

Les méthodes suivantes sont utilisées (voir [Bje88]) :

- la simulation directionnelle ou DS : mais la qualité des résultats se dégrade si la géométrie du domaine d'intégration est différente d'une hypersphère,
- les tirages d'importance ou IS (pour *Importance Sampling*) permettent de privilégier les directions susceptibles d'apporter le maximum d'informations.

3.2.3 Tirage hypercube latin (LHS)

La méthode de tirage hypercube latin, dite LHS (pour *Latin Hypercube Sampling*), fait partie des méthodes de stratification (ou *Stratified Sampling*).

Pour effectuer N tirages, le principe est le suivant :

- l'intervalle de variation de chaque variable aléatoire $\mathbf{x_i}$ est partagé en N segments équiprobables,
- on tire aléatoirement une valeur dans chacun de ces segments,
- pour une direction $\mathbf{x_i}$ donnée, on répartit les coordonnées des N tirages dans chacun des segments.

Une illustration est proposée avec la figure IV.4 pour deux lois normales centrées réduites ($\sigma = 1$) : les cercles isovaleurs à $\sigma = 1, 2, 3$ sont représentés. A gauche on voit un tirage de Monte-Carlo et à droite un tirage hypercube latin, avec la structure stratifiée en pointillés.

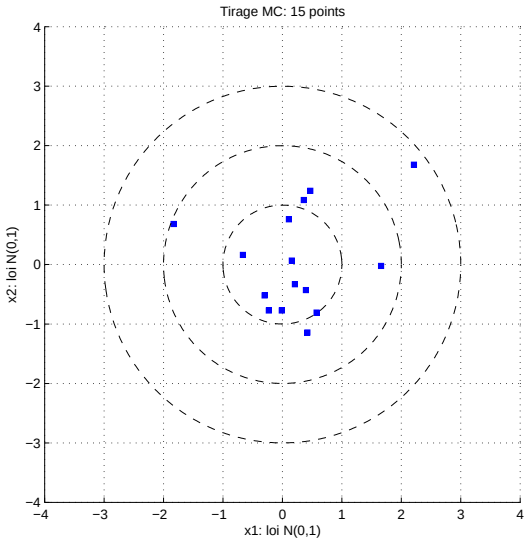
Le tirage hypercube latin assure une entière couverture du domaine de variation de X, c'est-à-dire que même des valeurs peu probables seront représentées, ce qui n'est pas assuré avec la méthode de Monte-Carlo, surtout si N n'est pas très grand. Le nombre de tirages requis est alors bien plus faible que pour la méthode de Monte-Carlo pour un niveau de convergence identique.

Il impose aussi un certain écart entre les points et assure une bonne répartition des tirages selon chaque direction x_i.

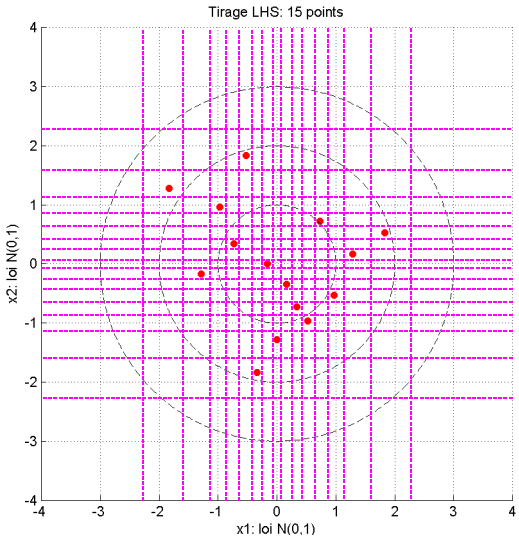
Dans le cas de lois uniformes, on retrouve le plan d'expérience dit **carré latin**, qui optimise un critère géométrique (voir la section 1.3.6 page 192).

3.3 Méthodes spectrales : décomposition de G

L'idée des méthodes spectrales est de décomposer la fonction de modélisation sous la forme d'un développement qui prend en compte l'effet des variables sous forme hiérarchique.



(a) Monte-Carlo



(b) Hypercube Latin

FIG. IV.4 – Comparaison tirages Monte-Carlo et Hypercube Latin en dimension 2 (lois normales)

La méthode HDMR (pour *High-Dimensional Model Representation*, voir [RASS99]) représente la fonction G sous la forme

$$G(X) = G_0 \ + \sum_{j=1}^{n} G_j(x_j) \ + \sum_{1 \leqslant i < j \leqslant n} G_{ij}(x_i, x_j) \ + \ldots \ + G_{12\ldots n}(X) \qquad \text{(IV.19)}$$

avec

- $G_0 \in \mathbb{R}$ l'effet moyen ou d'ordre 0,
- G_j correspond à l'effet de la variable isolée x_j,
- $G_{i_1 \ldots i_p}$ correspond à l'effet des combinaisons des variables $x_{i_1} \ldots xi_p$.

Cette décomposition est exacte et finie, mais pour la mise en oeuvre, il faut choisir une base de fonctions et calculer les coefficients qui permettent de décomposer les termes de la somme HDMR dans cette base : ce développement est alors infini, et on obtient une approximation de la fonction de modélisation G en choisissant un nombre fini de fonctions de base.

Le nombre d'évaluations du modèle nécessaires augmente cependant fortement avec la taille du problème (voir [Bou04]), à cause du nombre de coefficients à calculer. Se pose alors la question de la troncature de la décomposition. Il faut à la fois limiter le coût de calcul, mais aussi prendre assez de termes pour expliquer l'essentiel des variations de G. En pratique l'ordre 2 est souvent utilisé.

On peut éventuellement vérifier que l'approximation est satisfaisante *a posteriori* en calculant une erreur d'approximation.

Ensuite, on peut indiquer deux utilisations :

- Le modèle approché sert à effectuer une analyse statistique qui donne une estimation de la densité de probabilité de la sortie $G(\mathbf{X})$ ou les moments principaux : la démarche est similaire à celle des méthodes à surface de réponse dans l'approche probabiliste (voir section 4.2 page 171).
- En définissant des lois de probabilités pour les paramètres d'entrée, la variable aléatoire $G(\mathbf{X})$ se décompose de la même manière que la fonction de modélisation.

  Généralement, on travaille dans l'espace gaussien standard, et on utilise une base de polynômes orthogonaux (Legendre, Laguerre, Hermite,...) : ce développement est dit en "polynômes de chaos".

L'orthogonalité permet alors de décomposer la variance totale $Var[G(\mathbf{X})]$ et d'estimer facilement la contribution de chaque paramètre ou de combinaisons de ceux-ci (voir section 5.3.2 page 175).

Plusieurs algorithmes se proposent de calculer les coefficients du développement à coût réduit :

- DEMM (pour *Deterministic Equivalent Modeling Method*, voir [Tat95] et [Bou04]),
- SRSM (pour *Stochastic Response Surface Method*, voir [Isu99]).

Comme application de cette approche, on peut citer les éléments finis stochastiques ou SFEM (pour *Stochastic Finite Elements Method*, voir [GP91]) dans le domaine de la mécanique.

3.4 Méthodes FORM/SORM

Le lecteur peut se référer à [HM00], [DM96], [Bre84] ou [MKL86] à propos des méthodes présentées dans cette section.

Les méthodes FORM (pour *First Order Reliability Method*) et SORM (pour *Second Order Reliability Method*) ont été définies et sont principalement utilisées dans le domaine du calcul des structures. Elles ont été développées pour l'étude des petites

probabilités (inférieures à 10^{-2}) : nous avons vu que pour des domaines critiques de faible probabilité, les méthodes de tirage ne sont plus adaptées à cause du nombre de tirages nécessaires pour accéder à l'information dans cette zone.

Nous détaillons dans la suite le principe de ces méthodes, qui utilisent une approximation de G au voisinage du point de conception P^* pour déterminer le domaine de défaillance D_s.

3.4.1 Indice de fiabilité

Point de conception La première étape consiste à effectuer la transformation isoprobabiliste T pour travailler dans l'espace gaussien standard (voir section 2.1.6 page 155).

La deuxième étape consiste à trouver un point de conception P^* en résolvant le problème d'optimisation sous contrainte (IV.11) (voir page 157) que nous rappelons.

$$(P_2) \quad P^* = \min_{U \in \overline{D_s}} U^T U. \tag{IV.20}$$

Dans la section 4 sont présentées des méthodes pour résoudre le problème précédent.

Indice de fiabilité Ensuite on calcule l'indice de fiabilité dit de Hasofer et Lind (voir [HL74] pour la première définition).

Dans l'espace gaussien standard, l'indice de fiabilité β est défini comme la distance du point de conception à l'origine ou point nominal. On a donc

$$\beta = \|OP^*\|. \tag{IV.21}$$

Remarques

- Cet indice fournit une mesure de fiabilité comparative, intégrant l'incertitude des données sous une forme approximative, c'est-à-dire sans devoir complètement connaître leur distribution de probabilité mais seulement quelques unes de ses caractéristiques (typiquement, la moyenne et l'écart-type).
- Géométriquement, il correspond à la distance la plus petite entre l'origine et la zone de défaillance.
- Par convention, β est positif si l'origine O appartient au domaine de sûreté (c'est la cas habituel, voir section 2.1.5 page 154), sinon β est négatif.

La figure IV.5 représente une illustration du point de conception suivant deux directions dans l'espace gaussien standard, dans le cas d'un indice de fiabilité faible ($\beta < 3$), c'est-à-dire un domaine de défaillance proche du point nominal :

- la zone avec des hachures horizontales représente les points de distance inférieure à 3 σ de l'origine : ce support représente 99% de probabilité de réalisations de la loi gaussienne centrée réduite de densité Φ et écart-type $\sigma = 1$,

- la zone avec des hachures verticales représente la zone de défaillance $\overline{D_s}$ associée au seuil s,
- la probabilité p_s de dépasser le seuil s se calcule comme l'intégrale de la densité de probabilité Φ sur l'intersection de son support avec $\overline{D_s}$,
- la zone avec des hachures dans les deux sens représente donc le domaine contribuant à l'essentiel de la valeur de p_s.

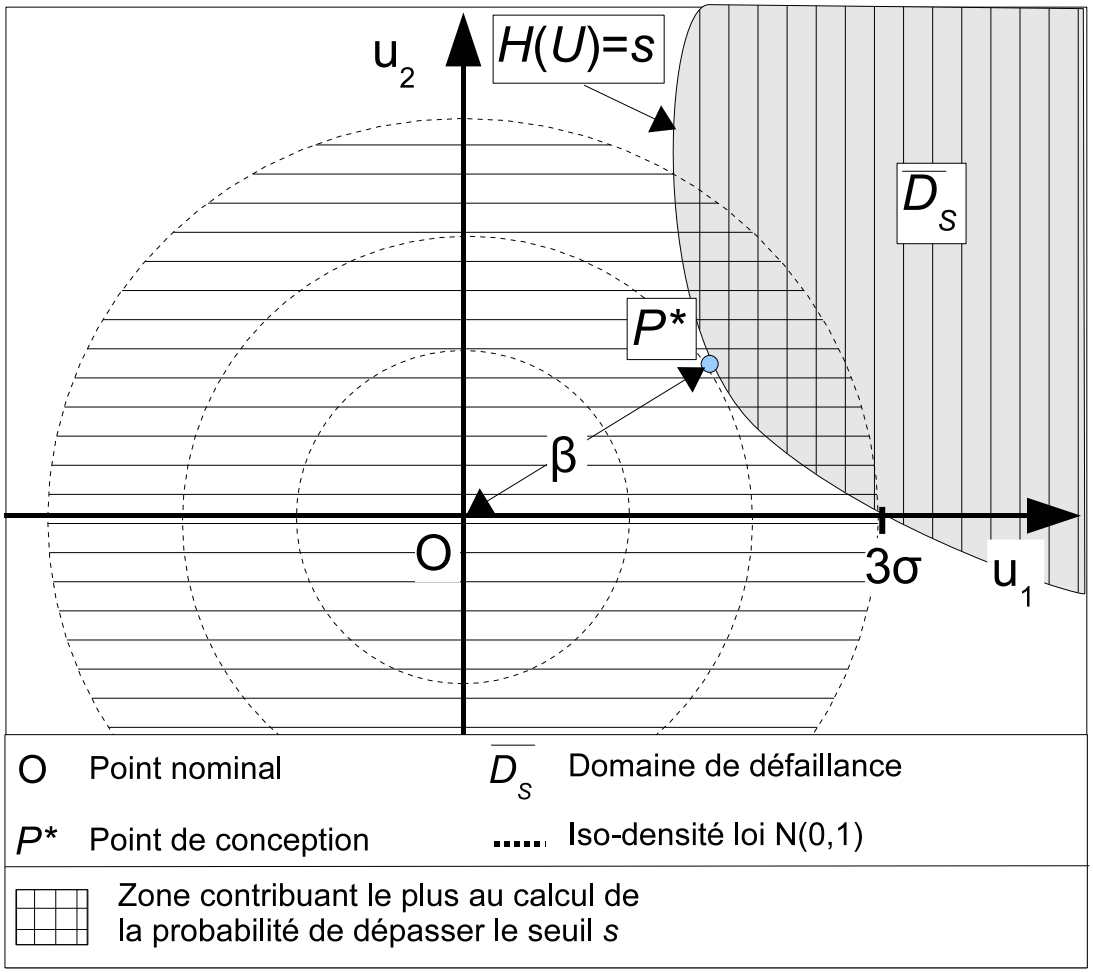


FIG. IV.5 – Point de conception, indice de fiabilité et probabilité de défaillance pour deux paramètres

Probabilité de défaillance : FORM et SORM Dans l'espace gaussien standard, soit le domaine de défaillance $\overline{D_s} := \{U \in \mathbb{R}^n;\ s \leqslant H(U)\}$ ne contenant pas O.

Le principe des méthodes FORM et SORM est d'exploiter l'idée suivante :

- La densité de probabilité notée ρ de la loi gaussienne centrée réduite décroît rapidement lorsque on s'éloigne de l'origine.

Cependant, cette propriété est de moins en moins vérifiée quand le nombre de paramètres augmente.

- P^* est le point de densité de probabilité maximale, et la contribution des autres points du domaine de défaillance $\overline{D_s}$ est inférieure à celle de P^*.

Ainsi,comme la probabilité de défaillance p_s à calculer est l'intégrale de ρ sur $\overline{D_s}$, on ne commet pas une grande erreur en remplaçant autour de P^* la fonction H par une approximation.

On obtient les relations suivantes (voir [Rac79]) :

- Pour la méthode FORM : on utilise une approximation de H du premier ordre, c'est-à-dire on la remplace par son hyperplan tangent au point P^* (voir figure IV.6).

  La probabilité de défaillance sur le domaine approché se calcule très simplement et on obtient

  $$p_s \simeq \Phi(-\beta) = 1 - \Phi(\beta) \tag{IV.22}$$

  avec $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-t^2}{2}}\ dt$
  la fonction de répartition d'une loi gaussienne centrée réduite.

  Dans [ZO99] est précisé que la méthode FORM n'est précise que dans le cas où le rayon de courbure de G est très grand et G définie par un petit nombre de paramètres aléatoires.

- Pour la méthode SORM : on utilise une approximation de G du second ordre.

  On approche la géométrie de G au voisinage de P^* par une surface contenant plus d'information : cela permet de dépasser les limitations de la méthode FORM.

  On obtient la formule suivante pour p_s

  $$p_s = \Phi(-\beta) \prod_{i=1}^{n} \frac{1}{\sqrt{1+\beta\kappa_i}} \tag{IV.23}$$

  avec κ_i les courbures principales de G au point P^*.

Remarque Dans le cas où la transformation isoprobabiliste T n'est pas unique, le choix conduit à des indices de fiabilité différents mais à la même probabilité de défaillance.

4 Méthodes de calcul des points de conception

4.1 Optimisation sous contraintes

4.1.1 Formulation

La recherche du point de conception (voir formule (IV.11) page 157) est un problème d'optimisation, que l'on peut formuler aussi sous contrainte d'inégalité classique :

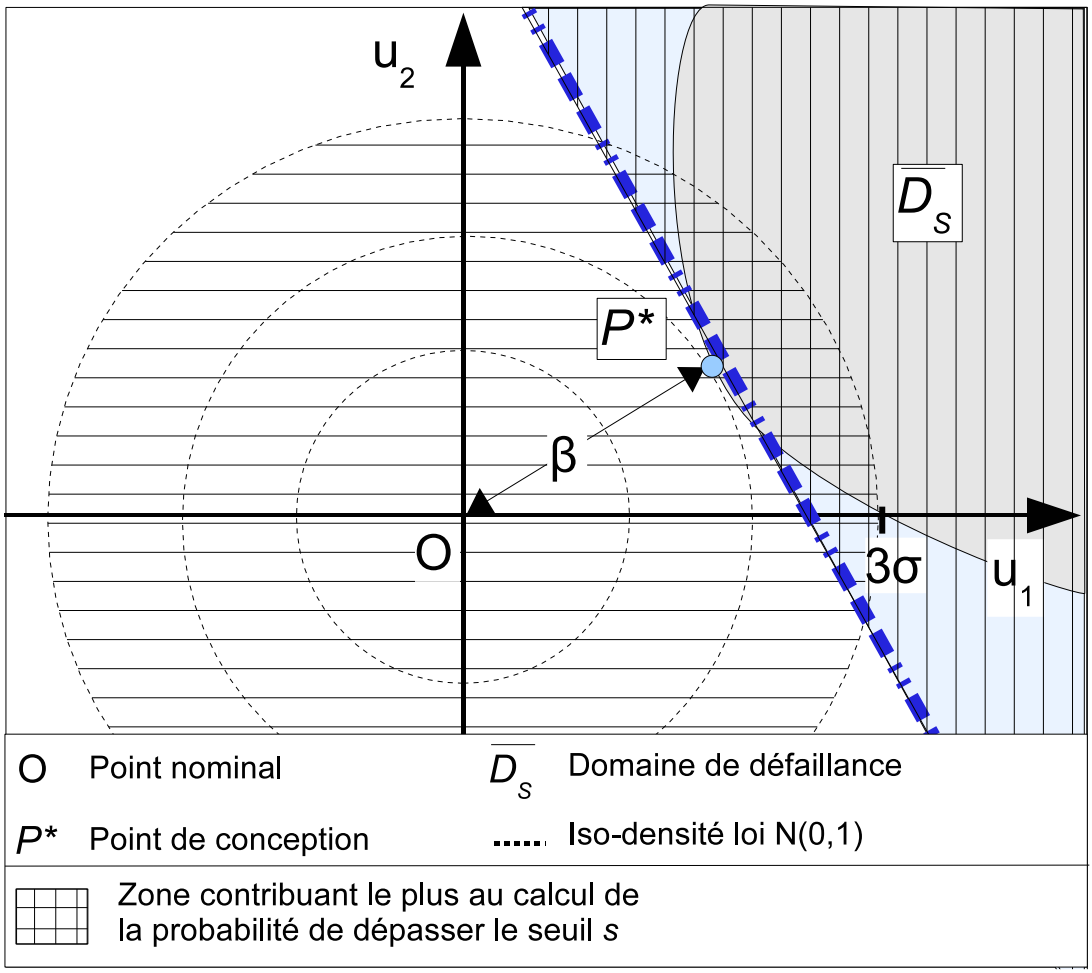


FIG. IV.6 – Méthode FORM : point de conception, indice de fiabilité et probabilité de défaillance pour deux paramètres

$$(P_2) \quad \begin{cases} P^* = \min U^T U \\ s \leqslant G(T^{-1}(U)). \end{cases} \tag{IV.24}$$

Le point optimal sans contrainte est l'origine O, donc dans le cas habituel où O n'appartient pas au domaine de défaillance (sinon la solution est trivialement O), la contrainte d'inégalité est toujours active, c'est-à-dire $s = G(T^{-1}(P^*))$. On peut alors, pour utiliser des méthodes d'optimisation sous contrainte d'égalité, formuler aussi le problème

$$\begin{cases} P^* = \min U^T U \\ s = G(T^{-1}(U)). \end{cases} \tag{IV.25}$$

4.1.2 Mise en oeuvre

Minima locaux Quand la géométrie du domaine de défaillance implique l'existence de plusieurs minima locaux, il n'existe pas de méthode robuste et efficace pour

trouver la solution, car l'optimisation globale est un problème en général difficile.

Dans ce cas, il faut trouver des stratégies pour se dégager d'un minimum local et trouver les autres minima ou le minimum global.

Par exemple dans [DKD98] est proposée une méthode de perturbation illustrée par la figure IV.7 et les étapes suivantes :

- trouver un minimum P_1^* en résolvant le problème (P_2) du point de conception,
- appliquer une perturbation à G de manière à éloigner P_1^* de l'origine O,
- trouver un autre minimum P_2^* par résolution du problème perturbé,
- répéter la perturbation en P_2^* et recommencer le processus.

  Si on converge vers un minimum dont la distance à O est supérieure à celles des points P_i^* précédents, c'est un minimum artificiel à rejeter et on arrête.

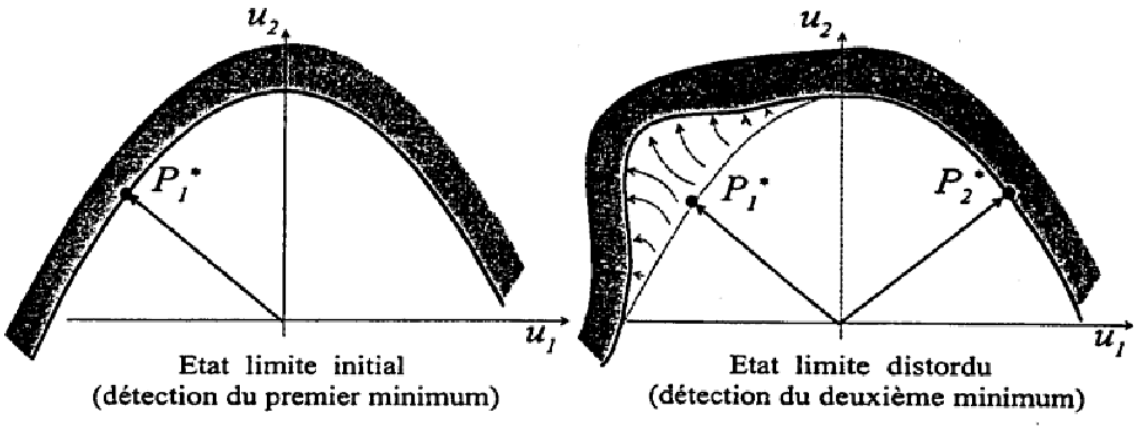


FIG. IV.7 – Méthode des perturbations pour plusieurs points de conception (figure extraite de [Lem05])

Difficultés La fonction à minimiser est élémentaire, mais le domaine de contrainte définissant les points admissibles est le domaine de défaillance $D_s := \{X \in \mathbb{R}^n;\ s \leqslant G(X)\}$, qui est défini de manière implicite.

Ainsi ce problème devient difficile quand la fonction de modélisation G présente une des caractéristiques suivantes :

- G n'est pas connue analytiquement mais est définie par un code de calcul,
- G n'est pas différentiable (cas précédent),
- G n'est pas régulière (par exemple elle présente des défauts de convexité).

Algorithmes Pour la recherche du point de conception, les algorithmes d'optimisation sous contraintes utilisés en pratique sont des méthodes itératives, et se classent de la manière suivante (voir [BGLS03], [Min83], [Lem05]) :

- Les **méthodes d'ordre zéro** ne font appel qu'à la fonction de modélisation G :
  - méthode de Powell (à base de dichotomies unidimensionnelles),
  - méthode de Nelder-Mead (ou polytopes mouvants).
- Les **méthodes d'ordre un** font appel à G et à son gradient ∇G :
  - méthode du gradient projeté,
  - méthodes des pénalités,
  - méthodes de points intérieurs,
  - méthodes du Lagrangien augmenté.

  L'algorithme HLRF (pour Hasofer-Lind-Rackwitz-Fiessler, voir [Rac79]), est une adaptation spécifique à la recherche du point de conception d'une méthode d'ordre un.
- Les **méthodes d'ordre deux** font appel à G, à ∇G, et à la hessienne $\nabla^2 G$
  - méthodes de Newton,
  - méthode SQP pour programmation quadratique séquentielle.
- Les méthodes d'ordre deux avec **approximation de la hessienne**, car le coût de calcul de la hessienne est souvent prohibitif
  - méthodes de quasi-Newton (Gauss-Newton, ...),
  - méthodes DFP (pour Davidon-Fletcher-Powell) et BFGS (pour Broyden-Fletcher-Goldfarb-Shanno).

Il est difficile de choisir la bonne méthode d'optimisation, car il faut prendre en compte plusieurs facteurs :

- type de fonction G (linéaire, quadratique, non régulière,...),
- possibilité de calculer le gradient ou la hessienne simplement,
- coût de calcul (lié au nombre de paramètres, au coût d'une évaluation de G, aux moyens de calcul,...).

Plus l'ordre de la méthode est élevé, plus la méthode est performante, c'est-à-dire qu'on diminue le nombre d'itérations nécessaires ; en contrepartie, le coût de calcul de chaque itération est augmenté.

Les méthodes d'ordre zéro ne sont utilisées que quand on ne peut pas faire autrement à cause de leur convergence très lente.

Les méthodes utilisant le gradient font partie des méthodes de descente, dont le principe est de calculer une direction de descente, puis le pas de descente optimal

dans cette direction pour passer du point X_k au point X_{k+1} en faisant décroître la fonction coût à minimiser.

Les méthodes d'ordre un sont plus performantes, et les méthodes d'ordre deux sont les plus efficaces, même si elles sont moins robustes (leur convergence n'est pas assurée), mais en pratique on peut rarement effectuer le calcul de la hessienne.

Les méthodes avec approximation de la hessienne semblent un bon compromis puisqu'elles intègrent une partie de l'information des dérivées secondes à moindre coût.

Des méthodes hybrides combinent plusieurs méthodes différentes pour essayer de bénéficier des avantages de chacune.

En général, on n'est pas assuré d'avoir trouvé le bon optimum car l'algorithme converge vers un point critique (gradient nul) qui peut être un minimum local, et il faut souvent faire appel à l'expertise de l'utilisateur pour valider le caractère global ou non de la solution.

4.2 Méthodes de surfaces de réponse

Cette classe de méthodes, dites RSM (pour *Response Surface Method*) consiste à construire une **approximation**, ou **surface de réponse**, notée g, de la fonction de modélisation G définie en section 2.1.1.

Ensuite, g est utilisée à la place de G avec les avantages suivants :

- Le **temps de calcul** des réponses par g est à présent **négligeable**.

  Une fois la surface de réponse construite, l'utilisateur peut évaluer le modèle approché sans se soucier du coût de calcul, et envisager d'utiliser des méthodes qui étaient écartées pour cette raison.

- Le **gradient** est ici **disponible**, et permet d'appliquer des méthodes d'optimisation efficaces utilisant cette information.

  Généralement, avec G non connue analytiquement, on peut calculer le gradient par différences finies ou différenciation automatique, mais on est confronté à des difficultés numériques ou à un coût prohibitif.

4.2.1 Approche choisie

Etant donné notre objectif, nous avons choisi d'utiliser la méthode de surfaces de réponse.

En effet, une fois l'approximation g construite, grâce aux deux avantages fondamentaux du coût de calcul négligeable et de la disponibilité du gradient, nous pouvons résoudre les trois problèmes $(P_i),\ i = 1, 2, 3$ à la fois :

- nous pouvons appliquer des méthodes de tirage pour calculer les probabilités de défaillance de (P_1),

- nous pouvons utiliser les algorithmes de descente (nécessitant le gradient) pour l'optimisation de (P_2),

- le gradient nous donne une information essentielle pour l'analyse de sensibilité de (P_3).

Ainsi, le résultat de notre étude dépendra de la qualité de notre approximation : le problème est difficile car nous voulons une méthode valable en grande dimension.

Le coût de calcul et la difficulté sont ainsi concentrés dans la construction de g, et nous devrons nous assurer de sa validité pour garantir la résolution de (P_i), $i = 1, 2, 3$ dans de bonnes conditions.

Il nous reste à choisir la méthode d'approximation appropriée : nous détaillons en section V les grandes classes de méthodes.

5 Méthodes pour déterminer les paramètres importants

5.1 Analyse de sensibilité

Comme indiqué dans la section 1.3.3 page 149, l'objectif de l'analyse de sensibilité est d'étudier :

- soit l'**influence des variations des paramètres** sur les variations de la réponse : la mesure de variation locale est alors de type **gradient**,
- soit l'**influence de la variabilité** des paramètres incertains, auxquels on associe ici une loi de probabilité, sur la variabilité de la réponse : une mesure de variabilité est la **variance**.

Le choix dépend du problème et de la notion de paramètre important que définit l'utilisateur du modèle. Dans les deux cas, l'analyse de sensibilité permet de déterminer les paramètres influents.

Méthodes De nombreuses techniques existent, le choix dépend du type de problème, des caractéristiques du modèle étudié, du coût de calcul autorisé (voir [SCS00]).

Une méthode d'analyse de sensibilité est basée sur la mesure de sensibilité choisie :

- Une méthode locale se donne pour objectif de calculer une dérivée partielle évaluée en un point. On étudie l'effet sur la réponse de petites variations sur un paramètre incertain en gardant les autres paramètres fixés.
- Une méthode globale mesure les critères présentés précédemment dans l'analyse de robustesse ou l'approche probabiliste.

5.2 Sensibilité locale

La sensibilité locale mesure l'influence locale des paramètres.

Cette approche porte sur le comportement de G sans prendre en compte les lois de probabilités des paramètres incertains (sauf si on calcule le gradient au point le plus probable). Cependant, dans le cas de modèles non linéaires avec un nombre important de paramètres, une méthode de sensibilité globale est mieux adaptée.

Principe On l'utilise par exemple en faisant varier chaque paramètre dans un petit intervalle autour d'un point. On suppose en général une dépendance linéaire entre la sortie et les entrées du modèle, et on choisit des amplitudes de variations égales (par exemple $+/-10\%$ autour du point).

En général on calcule des indices à partir du gradient en certains points considérés comme stratégiques par l'utilisateur, comme par exemple le point le plus probable.

Remarque Dans le domaine de la mécanique, des mesures spécifiques, dites "indices de sensibilité fiabiliste", ont été développées pour évaluer les effets des variations des paramètres sur la défaillance de la structure, donnée par l'indice de fiabilité β (voir section 3.4.1 page 165) au point de conception.

5.2.1 Gradient

Dans le cas où le gradient $\nabla G(X)$ est calculable, il mesure la sensibilité s_i de G par rapport aux variations des paramètres x_i :

$$s_i := \frac{\partial G(X)}{\partial x_i}. \tag{IV.26}$$

Normalisation La valeur d'une dérivée partielle $\nabla_i G(X)$ dépend des grandeurs numériques issues de l'intervalle de définition du paramètre x_i.

Afin de comparer les dérivées dans les différentes directions, il apparaît ainsi nécessaire de normaliser le résultat.

Par exemple on calcule l'indice $\overline{s_i}$ sous la forme

$$\overline{s_i} := \frac{\partial G(X)}{\partial x_i}\frac{x_i}{G(X)}. \tag{IV.27}$$

On peut aussi normaliser l'intervalle de définition $[a_i, b_i]$ de chaque paramètre avec le changement de variable

$$\begin{array}{ccc} [a_i, b_i] & \longrightarrow & \mathbb{R} \\ x_i & \longmapsto & \overline{x_i} := \frac{x_i - a_i}{b_i - a_i} \end{array} \tag{IV.28}$$

et calculer les gradients de la fonction normalisée $G(\overline{X})$ définie sur $[0,1]^n$.

Calcul pratique Plusieurs techniques permettent de calculer des dérivées partielles de la foncion de modélisation :

- la différenciation du problème continu peut s'effectuer
  - de manière analytique : cette méthode ne s'applique qu'à des modèles simples,
  - par approximation avec les différences finies : cette approche est coûteuse et sujette à des instabilités numériques,

- la différenciation du problème discret peut être menée de manière automatique : l'algorithme travaille sur le code source du modèle, dit code direct, et génère automatiquement un code dit dérivé qui permet le calcul du gradient en un point.

  Cette méthode permet de calculer des dérivées sans besoin d'avoir accès aux équations mais avec une plus grande efficacité que l'approximation par différences finies.

  Plusieurs logiciels de différenciation automatique existent, tels ODYSSEE (voir [FP98]) ou ADIFOR (voir [BCC+92]).

  Nous utiliserons une implémentation de différenciation algorithmique (le code source est dérivé en écrivant les instructions directement dans le code) dans le cadre des réseaux de neurones (voir section 2.3.4 page 215 du chapitre suivant).

5.2.2 *Screening*

La technique du *screening* est un cas particulier de sensibilité locale, adaptée aux modèles coûteux ou en grande dimension : elle consiste à effectuer des évaluations numériques du modèle à partir d'un plan d'expériences.

Souvent, elle est utilisée comme étude préliminaire afin d'identifier les paramètres importants : cette approche est présentée dans la section 5.4.2 sur les plans d'expérience, mais elle souffre du coût de calcul.

Exemple d'utilisation Une méthode dans un cas de grande dimension n a été proposée dans [Mor91] : le coût est alors proportionnel à n.

Le principe est de définir un effet élémentaire noté $d_i(X)$ suivant une direction x_i au point X comme un taux d'accroissement qui nécessite deux évaluations de la fonction de modélisation G.

Après un tirage aléatoire $\{X_k, k = 1 \dots N\}$, une distribution d'effets élémentaires $F_i := \{d_i(X_k), k = 1 \dots N\}$ suivant x_i est calculée.

La caractérisation de F_i par sa moyenne et sa variance empiriques donnent alors une indication sur l'influence du paramètre x_i.

5.3 Sensibilité globale

Cette approche se propose d'estimer la part d'incertitude de chaque paramètre sur l'incertitude totale en étudiant le comportement de la réponse à l'aide des densités de probabilités des paramètres d'entrée : elle est fondée sur le calcul d'indices représentant la contribution relative d'un ou plusieurs paramètre à la variance de la réponse $G(\mathbf{X})$.

Contrairement à l'approche précédente, on évalue l'effet d'un paramètre x_i

- avec des variations sur l'ensemble de l'intervalle de définition,
- alors que les autres paramètres varient aussi.

5.3.1 Méthodes de tirage

Dans cette catégorie apparaissent les méthodes de tirage vues en section 3.2 qui permettent d'estimer la moyenne et la variance de la réponse.

5.3.2 Méthodes de mesure d'importance : indices de Sobol

Cette classe de méthodes se propose d'estimer la quantité suivante

$$\frac{Var_{\mathbf{x_i}}[E(G(\mathbf{X}) \mid \mathbf{x_i} = x_i)]}{Var(G(\mathbf{X}))} \tag{IV.29}$$

avec

- $E(G(\mathbf{X}) \mid \mathbf{x_i} = x_i)$ l'espérance de $G(\mathbf{X})$ conditionnellement à une valeur fixée de $\mathbf{x_i}$,
- $Var_{\mathbf{x_i}}$ la variance sur toutes les valeurs possibles de $\mathbf{x_i}$.

Cette mesure est une variance conditionnelle qui estime la part du paramètre x_i dans la variance totale.

Elle peut manquer de robustesse et d'autres mesures d'importance ont été proposées (voir par exemple [IH90]).

Les indices de Sobol (voir [Sob93]) sont basés sur la décomposition de la fonction de modélisation G dans une base de fonctions orthogonales : par exemple sous forme de "chaos polynomial" (voir section 3.3 page 162).

L'orthogonalité implique la décomposition de la variance de $G(\mathbf{X})$, notée V_G, sous la forme des sommes des variances des fonctions de base

$$V_G = \sum_{j=1}^{n} V_j \; + \sum_{1 \leqslant i < j \leqslant n} V_{ij} \; + \ldots \; + V_{12\ldots n} \tag{IV.30}$$

avec n le nombre de variables de la fonction.

On définit ensuite les indices de sensibilité de Sobol

- au premier ordre avec
$$S_i := \frac{V_i}{V_G}, \tag{IV.31}$$
- à un ordre supérieur avec
$$S_{i_1 i_2 \ldots i_p} := \frac{V_{i_1 i_2 \ldots i_p}}{V_G}. \tag{IV.32}$$

La somme de tous les indices vaut un, et chaque indice représente ainsi une fraction de la variance de la réponse

- S_i mesure la part expliquée par le paramètre x_i (premier ordre), et constitue une estimation de la mesure d'importance (IV.29) ci-dessus.

- $S_{i_1 i_2 \ldots i_p}$ mesure la part expliquée par la combinaison des paramètres correspondants $x_{i_1} x_{i_2} \ldots x_{i_p}$ (ordre supérieur).

Le nombre de termes dans la somme (IV.30) est égal à 2^n, mais habituellement on se restreint aux termes d'ordre inférieur pour des fonctions de modélisation qui ne présentent que de faibles intéractions entre les variables.

Chaque terme $S_{i_1 i_2 \ldots i_p}$ peut être calculé

- à partir d'une intégrale multidimensionnelle, il faut alors utiliser une méthode de quadrature numérique (par exemple la méthode de Monte-Carlo),
- par une relation simple avec le coefficient correspondant de la décomposition de G, qui a été calculé auparavant avec une méthode spécifique (voir section 3.3 page 162).

Cette approche permet une analyse plus fine des effets conjugués de plusieurs paramètres, mais une limitation de cette approche est le coût de calcul élevé, notamment pour des modèles non linéaires pour lesquels il faut prendre en compte des indices d'ordre élevé.

La méthode FAST (pour *Fourier Amplitude Sensitivity Test*, voir [CLS78]) permet de calculer les indices de Sobol du premier ordre avec un coût réduit.

5.4 Plans d'expérience

Les plans d'expérience (voir leur présentation en section 1.3) peuvent être utilisés directement pour une analyse de sensibilité sans avoir recours au gradient.

5.4.1 Plans à deux niveaux

Ces plans faisant intervenir deux niveaux servent à évaluer l'amplitude de la réponse par exemple pour un paramètre entre ses bornes inférieure et supérieure.

Dans le plan factoriel complet à deux niveaux, toutes les valeurs extrêmes (bornes -1 et 1) interviennent, mais le nombre d'expériences 2^n croît de manière exponentielle avec la dimension.

Des plans factoriels fractionnaires permettent de réduire le coût, mais ces méthodes deviennent aussi impraticables quand la dimension n du problème augmente.

5.4.2 *One parameter at a time*

Comme son nom l'indique, cette approche consiste à faire varier un seul paramètre à la fois (par exemple dans [DHV92]) et est utilisée dans le cadre du *screening* (voir section 5.1 page 172) afin d'identifier les paramètres importants :

- on considère trois niveaux pour tous les paramètres (par exemple le milieu de leur intervalle de définition, et les deux bornes),
- on fixe la valeur de tous les paramètres sauf un, par exemple au deuxième niveau (le milieu),

- on évalue le paramètre libre aux trois niveaux pour connaître les variations de la réponse.

Cette méthode revient à utiliser un plan d'expérience à trois niveaux : elle suppose implicitement que les paramètres n'entrent pas en intéraction les uns par rapport aux autres et permet d'évaluer les effets dits principaux (sans intéraction).

En utilisant des plans factoriels, on peut évaluer les effets du premier ordre (avec intéractions), mais le nombre de niveaux augmente, et le nombre de points nécessaires devient excessif en grande dimension.

Chapitre V

Méthodes d'approximation

Sommaire

1 Introduction

1.1 Hypothèses

Nous reprenons les notations de la section 2.1 page 152 du chapitre précédent, et nous posons ici les hypothèses pour la suite de notre étude.

1.1.1 Approximation

Nous voulons construire une fonction g, approximation du modèle initial G

$$g : \quad \Omega \subset \mathbb{R}^n \quad \longrightarrow \quad \mathbb{R}. \tag{V.1}$$

Pour utiliser g à la place de G et résoudre plus facilement le problème initial, g possède les propriétés suivantes :

- g est une fonction analytique dont l'évaluation est peu coûteuse,
- le gradient de g est facilement calculable (ainsi on peut utiliser g pour un problème d'optimisation).

Dans notre travail, nous appellerons g **approximation** ou **modèle approché** (on trouve aussi le terme de surface de réponse).

Pour le problème de construction de l'approximation, on trouve aussi le terme de problème d'estimation, et en statistique de modélisation de données.

1.1.2 Données

Nous disposons de données :

- un échantillon de points noté $D := \{X_1, \ldots, X_N\} \subset \Omega$ de l'espace des paramètres de départ,
- le modèle est évalué pour obtenir les valeurs $\{G(X_1), \ldots, G(X_N)\} \subset \mathbb{R}$.

Nous considérons que nous pouvons choisir le nombre (dans la mesure d'un coût raisonnable) et l'emplacement de ces données, dites **données de construction**, car elles servent à calculer g.

Si un processus itératif est ensuite utilisé pour construire g, on les appelle aussi **données d'apprentissage**.

Remarque : cadre non déterministe Ainsi, travailler avec un code de simulation procure un avantage important par rapport aux modèles basés sur des expériences réelles. En effet dans ce dernier cas, les données non seulement peuvent être bruitées (deux expériences avec les mêmes paramètres peuvent produire deux résultats différents), mais des contraintes multiples peuvent aussi avoir pour conséquence des données limitées en nombre, ou des expériences interdites, ce qui rend le problème plus difficile.

Dans cette approche, on recueille des données qui correspondent uniquement à des mesures ou à des expériences d'un système sans disposer de modèle initial, avec

- Un ensemble de points $\{X_1, \ldots, X_N\} \subset \Omega$: il peut être choisi avec un plan d'expérience ou imposé par la nature du problème.
- Les mesures $\{y_1, \ldots, y_N\} \subset \mathbb{R}$ associées aux X_i et correspondant à un critère du système étudié : ici ces mesures ne sont pas des simulations, elles sont non déterministes et donc souvent entachées d'erreurs (ou bruit).

On suppose alors qu'il existe une relation qu'on peut décrire sous forme mathématique $g(X)$ entre les variables X et les mesures y : on cherche à construire un modèle d'approximation g qui rend au mieux compte du phénomène, sans étudier les équations du système (on peut être dans l'impossibilité de le faire), uniquement avec l'information fournie par les donnnées dont on dispose.

On parle alors de "lissage", ou "modélisation boîte noire", par opposition aux modèles mathématiques établis après une analyse physique, économique,... suivant le cas.

En adoptant une approche statistique, pour un point X_0 donné, la valeur $g(X_0)$ cherchée correspond au calcul de la moyenne sur un très grand nombre de mesures $\mathbf{y_0}$ (la mesure de l'expérience est vue comme une variable aléatoire) associées à X_0 : c'est l'espérance de $\mathbf{y_0}$ pour la valeur X_0, et la fonction g s'apelle "fonction de régression" (voir section 1.2.4 page 185)

Dans l'approche boîte noire et statistique, g est appelé simplement "modèle".

Erreur et résidus Nous appelons **vecteur des résidus** noté R, défini par les écarts entre le modèle G et l'approximation g sur les données de construction D :

$$R := (G(X_1) - g(X_1), \ldots, G(X_N) - g(X_N))^T. \tag{V.2}$$

Nous définissons l'**erreur d'approximation** e_D comme la mesure des résidus

$$e_D := \|R\| \tag{V.3}$$

avec $\|.\|$ une certaine norme à préciser.

En prenant les résidus sur un autre ensemble que les données de construction, on définit un autre type d'erreur, dite **erreur de vérification**.

Qualité de prédiction A partir des informations fournies par les données, g doit assurer une prédiction des valeurs de G sur l'ensemble du domaine Ω, c'est-à-dire pour des points hors des données de construction

$$\forall X \in \Omega \quad G(X) \simeq g(X). \tag{V.4}$$

Cette propriété, appelée **qualité de prédiction**, ou de **généralisation** (en statistique on trouve le terme d'"inférence"), est fondamentale pour pouvoir commettre des erreurs négligeables ou faibles lors de l'utilisation de g en substitution de G, car une approximation qui généralise mal n'est pas représentative du modèle initial.

Nous nous plaçons donc dans le cadre de l'**approximation globale** d'une fonction, d'autres techniques servent à construire une approximation locale, valable uniquement au voisinage d'un point (développement en série de Taylor par exemple).

1.1.3 Objectif

La stratégie de remplacer un modèle coûteux par un modèle approché analytique non coûteux a évidemment de nombreuses applications dans tous les domaines du calcul numérique.

En effet, malgré la progression constante de la puissance de calcul, le coût de calcul d'une simulation numérique d'un phénomène complexe reste élevé (de quelques heures à plusieurs dizaines d'heures pour le calcul des effets d'un crash automobile ou du profil aérodynamique d'un avion). Et dans la plupart des cas, le recours direct au modèle n'est pas envisageable pour l'analyse : *design*, optimisation, fiabilité,...

L'idée est alors de construire une approximation mathématique simplifiée et peu coûteuse pour l'utiliser en remplacement du modèle. Cette approche se rencontre sous les termes de "*metamodel*", "*surrogate model*", ...

Il existe ainsi de nombreuses méthodes d'approximation disponibles dans ce cadre.

Dans notre cas, notre choix est guidé par la contrainte d'un grand nombre de paramètres d'entrée incertains, et le fait que nous n'avons aucune information à priori sur la régularité de la fonction G. La méthode de construction de g doit ainsi tenir compte des exigences suivantes :

- un nombre réduit d'évaluations de G à cause de la grande dimension,
- g doit permettre de prédire la réponse du modèle G pour des points non encore évalués de l'ensemble du domaine de définition des paramètres,
- la méthode doit s'appliquer à des fonctions non régulières.

1.2 Construction de l'approximation

1.2.1 Principe

Construire une approximation g d'un modèle numérique implique généralement les étapes suivantes (voir par exemple [SBG+02]) :

- choix des données de construction, c'est-à-dire l'ensemble des points où le modèle sera évalué pour apporter de l'information,
- choix de la nature de g,
- construction de g par ajustement avec les données,
- validation de g par le calcul d'une erreur sur des données de vérification.

Les deux premiers choix sont liés, et l'utilisateur doit chercher un compromis entre la meilleure précision pour son approximation et le coût de calcul déterminé par le nombre de points de construction et la méthode de calcul de g.

1.2.2 Choix des données de construction

Le choix des données est extrèmement important dans la construction de g, d'une part car il faut économiser le nombre d'évaluations du modèle, et d'autre part car la répartition des points a une influence importante sur la précision de l'approximation.

Elles peuvent provenir selon la méthode choisie

- D'un ensemble de points fixé à l'avance de manière déterministe (par exemple formules de quadrature pour l'interpolation polynomiale).

- D'un plan d'expériences, qui consiste à placer de manière efficace les donnés dans l'espace des paramètres (voir section 1.3).

  L'utilisateur choisit parmi les plans existants ceux qui sont recommandés en fonction de l'objectif poursuivi (nombre de points, nature de l'approximation,...).

  Les plans d'expériences utilisés dans le cadre de la simulation numérique pour contruire une approximation s'appelllent "*space filling designs*" : ils sont de type carré latin, uniforme, minimax et maximin,...

1.2.3 Choix de la nature de g

Le choix de la nature de g dépend de l'objectif poursuivi, des hypothèses de régularité sur G, et détermine la méthode à mettre en oeuvre.

Le lecteur trouvera dans [MM95] un exposé de la méthode des surfaces de réponse.

En termes mathématiques, on cherche à minimiser la distance entre les fonctions G et g : le théorème de projection orthogonale nous donne l'existence de g qui réalise la plus courte distance de G au sous-espace $\mathcal{A}$ auquel appartient g

$$\forall G \in \mathcal{E}\ \exists g \in \mathcal{A} \quad \|G - g\| = \min_{h \in \mathcal{A}} \|G - h\| \tag{V.5}$$

avec

- $\mathcal{E}$ espace fonctionnel normé de dimension infinie,
- $\mathcal{A}$ sous-espace vectoriel de dimension finie,
- $\|\|$ une norme à préciser.

L'unicité de l'approximation g est généralement un problème difficile, mais si $\mathcal{E}$ est un espace de Hilbert, alors l'unicité de g est assurée.

Exemple Le théorème précédent s'applique par exemple en dimension un avec

- $\mathcal{E} = \{\mathcal{C}([a,b])\}$ les fonctions continues sur le segment $[a,b]$,
- $\mathcal{A} = \mathcal{P}_n$ les polynômes de degré inférieur ou égal à n sur $[a,b]$,
- $(f,g) = \int_a^b f(x)\ g(x)\ dx$ le produit scalaire et la norme L_2 associée.

On a aussi le développement en série de Fourier des fonctions périodiques.

Trouver la meilleure approximation dépend alors de la topologie des espaces mis en oeuvre (choix de la norme et choix de $\mathcal{A}$).

L'approximation s'obtient donc par **projection sur une base finie de fonctions** qui peuvent être :

- des polynômes,
- des fonctions radiales,

- des ondelettes, ...

Si la base de l'espace d'approximation $\mathcal{A}$ est $W = \{w_1, \ldots, w_p\}$ alors la projection g s'écrit sous la forme très générale

$$g(X) = \sum_{i=1}^{p} b_i \; w_i(X) \tag{V.6}$$

avec les coefficients $B = (b_1, \ldots, b_p)$ à déterminer.

On cherche aussi à obtenir des résultats

- de convergence : l'erreur $\|G - g\|$ tend vers 0 quand le nombre de fonctions de base tend vers l'infini, et selon la norme choisie on obtient la convergence uniforme, quadratique,...
- et de majoration d'erreur, qui permettent de contrôler la qualité de l'approximation.

1.2.4 Construction de g

Une fois la nature de l'approximation g choisie, g est définie à l'aide d'un certain nombre de coefficients $B \in \mathbb{R}^p$ et nous noterons $g(X, B)$ quand les coefficients doivent être calculés.

On utilise l'information fournie par les données afin de calculer B de telle façon que les réponses de G et de g soient proches : c'est la **phase d'ajustement**.

Dans la suite, on présente les différentes approches utilisées.

Minimisation de l'erreur On peut minimiser la norme de l'erreur de l'approximation : l'ajustement se ramène alors au problème d'optimisation

$$\min_B \; \|R_B\|^2 \tag{V.7}$$

avec $R_B = (G(X_1) - g(X_1, B), \ldots, G(X_N) - g(X_N, B))$ le vecteur des résidus sur les données de construction.

Cette méthode est très générale et s'applique à tout type d'approximation :

- Avec la norme infinie, on calcule une approximation dite uniforme,
- Avec la norme euclidienne, on calcule une approximation dite des **moindres carrés**.

  C'est la méthode la plus courante, qui consiste donc à résoudre

$$\min_B \; \sum_{i=1}^{N} (G(X_i) - g(X_i, B))^2. \tag{V.8}$$

Interpolation L'interpolation consiste à faire passer l'approximation par les données :

$$\forall i = 1, \ldots, N \quad g(X_i, B) = G(X_i). \tag{V.9}$$

Cette méthode s'applique généralement à l'approximation par polynômes sur un ensemble de points (ou noeuds) définis analytiquement : on n'utilise pas de plan d'expérience (voir section 1.4.2).

Les *sparse grids* constituent une forme d'interpolation multivariée présentée dans la section 3.

Krigeage Le krigeage sert à construire une approximation avec une estimation de la variance de l'erreur.

Plusieurs variantes existent, les plus connues interpolent les données, mais d'autres servent à lisser des données bruitées sans les interpoler (voir section 1.5.3).

Régression On trouve le terme de régression quand la réponse notée ici $\mathbf{Y}$ du modèle est une variable aléatoire : pour un point X donné de l'espace des paramètres d'entrée, $\mathbf{Y}(X)$ se répartit suivant une certaine distribution.

Nous sommes dans le cadre d'un modèle non déterministe, dont la réponse est issue de mesures bruitées : les expériences sont non reproductibles.

La fonction ou modèle de régression $g(X, B)$ donnant une réponse unique pour toute valeur de X, sa prédiction est donc entachée d'erreur.

Le problème s'écrit alors

$$\mathbf{Y} = g(X, B) + \varepsilon \tag{V.10}$$

avec ε une variable aléatoire définissant l'erreur entre l'approximation et le modèle, telle que $E(\varepsilon) = 0$: en général, les méthodes de régression utilisent l'hypothèse simplificatrice que la variance de ε ne dépend pas de X (homoscédasticité).

La fonction de régression g est celle qui, pour tout échantillon $\{X_1, \ldots, X_N\}$ de X produit la valeur moyenne de $\mathbf{Y}(X)$. Ainsi g matérialise l'espérance de $\mathbf{Y}$ conditionnellement à X

$$g(X, B) = E(\mathbf{Y}|X). \tag{V.11}$$

Généralement, on utilise un polynôme de degré un ou deux pour g (la régression linéaire repose sur l'hypothèse la plus simple, selon laquelle g est une fonction linéaire), mais on peut aussi choisir des fonctions de type splines, réseaux de neurones...

Ensuite, il existe deux méthodes principales pour ajuster les coefficients B aux données :

- La méthode des moindres carrés (voir ci-dessus) : B minimise la somme des carrés des erreurs de prédiction sur les données disponibles (ou norme au carré des résidus), c'est-à-dire
  $\sum_{j=1}^{N} \varepsilon_j^2$.
  Dans la méthode des "moindres carrés linéaires", on considère la linéarité par rapport aux coefficients B, et non par rapport aux variables (ainsi pour un

modèle de régression polynomial $y = b_0 + b_1\, x + b_2\, x^2 + \ldots + b_p\, x^p$ linéaire par rapport aux b_j) : on résoud alors un système linéaire.

L'utilisation d'un modèle de régression non linéaire par rapport à B, méthode des "moindres carrés non linéaires", produit un système d'équations non linéaires que l'on résoud par une méthode d'optimisation.

- La méthode du "maximum de vraisemblance" : B rend maximale la vraisemblance, qui est une mesure de l'adéquation entre une distribution et un échantillon. Ici elle quantifie l'erreur entre la distribution des données $G(X_i)$ et de l'approximation $g(X_i, B)$.

1.2.5 Qualité de prédiction

Sur-ajustement On demande à l'approximation de posséder une bonne qualité de prédiction, or la cause la plus fréquente de mauvaise généralisation est le sur-ajustement : cette erreur de conception consiste à construire une approximation incorporant un trop grand nombre de coefficients par rapport aux données.

On rencontre ce problème en statistique sous le terme de "dilemne biais-variance", nous le présentons sous forme d'exemple.

Soit une distribution de données fortement non linéaire,

- Un polynôme de faible degré n'aura pas le nombre de degrés de liberté nécessaires pour s'ajuster à la forme générale de la distribution des données : l'approximation commet des erreurs de prédictions importantes (biais important).

  Cependant, en raison même de cette rigidité, les prédictions du modèle vont dépendre peu du choix des données de construction (faible variance).

- Un polynôme de degré élevé parviendra à ajuster beaucoup mieux les données : l'erreur sur l'échantillon de construction est faible (faible biais).

  Mais l'approximation obtenue devient instable par rapport au choix des données : avec un échantillon différent, afin de réduire l'erreur sur les résidus, le polynôme de degré élevé va s'ajuster aux nouvelles données avec des coefficients très différents, et les prédictions seront complètement différentes (variance importante).

Dans le cas des réseaux de neurones, on parle de "sur-apprentissage" pour ce phénomène (voir section 2.2.1 page 2.2.1).

Problème mal posé Le phénomène de sur-ajustement vient du fait que le problème d'approximation fait partie des problèmes mal posés (voir [Kir96]) : en général, il ne peut pas être résolu de manière unique avec les données en nombre fini dont on dispose.

Par exemple, il existe une infinité d'approximations passant par un ensemble de points donnés (voir illustration avec l'interpolation en dimension un dans la figure V.1).

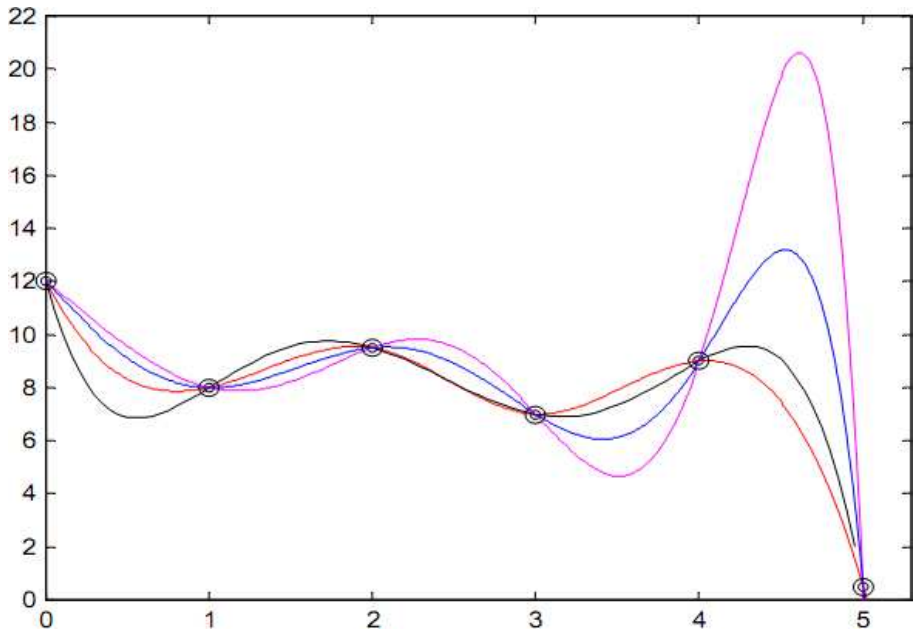


FIG. V.1 – En dimension un, une infinité de courbes passent par un ensemble de données

Si on ajoute des conditions, en précisant la nature de l'approximation (polynôme d'un certain degré par exemple), le problème peut devenir bien posé, si le nombre de données est adapté à la taille du vecteur des coefficients B à calculer.

En pratique, la plupart du temps le nombre de données est très supérieur aux nombre de coefficients à déterminer : le problème est sur-déterminé.

Pour remédier à ce problème

- on essaie de respecter la propriété de parcimonie (voir ci-dessous),
- et on fait appel à des techniques de régularisation (voir section 1.5.2 page 198).

Parcimonie La construction de la meilleure approximation requiert la recherche de sa complexité optimale :

- Une approximation avec peu de coefficients a une faible capacité de prédiction en raison de son biais important (pas assez de degrés de liberté).
- Une approximation avec trop de coefficients a une faible capacité de prédiction en raison de sa variance importante (trop grande sensibilité aux choix des données de construction et instabilité numérique).

On trouve aussi l'expression de **parcimonie** pour indiquer que l'on cherche à ce que le nombre de coefficients de l'approximation à ajuster soit le plus faible possible (voir [DMS+02]).

On montre (voir [Bar93]) que pour une précision donnée, le nombre de coefficients croît

- de manière exponentielle avec le nombre de variables si l'approximation est linéaire par rapport à ses coefficients B :

  par exemple les polynômes ou les projections sur des fonctions de base sont des approximations peu parcimonieuses,

- de manière linéaire avec le nombre de variables si l'approximation est non linéaire par rapport à ses coefficients B :

  par exemple les réseaux de neurones sont plutôt parcimonieux.

Ainsi le critère de parcimonie est fondamental pour réaliser une approximation à moindre coût en grande dimension :

- pour une précision donnée, le nombre de simulations à réaliser sera plus petit,
- ou pour un nombre d'expériences donné, l'approximation obtenue sera plus précise.

Il est plus facile d'éviter le phénomène de sur-ajustement avec une approximation parcimonieuse.

1.2.6 Validation de g

Une fois l'approximation construite, il faut la valider pour être assuré qu'on peut l'utiliser pour remplacer le modèle initial. On voit d'après ce qui précède qu'il faut accorder une grande importance à cette phase en testant la capacité de prédiction.

Une fois l'approximation g calculée avec les données de construction, il faut calculer un critère global pour juger les performances de g.

Cette validation est basée sur l'analyse des résidus sur des données de validation (différentes des données de construction).

Dans [JCS01] est présentée une étude de plusieurs métriques permettant de comparer les approximations suivant les critères de qualité souhaités

- la qualité de prédiction, ou de généralisation, est généralement l'objectif principal recherché,
- l'efficacité : elle dépend du coût requis pour construire l'approximation,
- la robustesse : la capacité d'obtenir une bonne précision pour différents problèmes ou taille de données,
- la simplicité de mise en oeuvre,...

Nous nous intéressons dans cette étude à la qualité de prédiction.

Données de validation On effectue donc un tirage noté $V := \{Y_1, \ldots, Y_M\}$ sur lequel on évalue le modèle G pour obtenir des **données de validation**, différentes des données de construction.

On peut utiliser un plan d'expérience.

Critères de validation Les critères usuels de performance sont alors

- l'erreur quadratique moyenne

$$E := \sqrt{\frac{1}{M} \sum_{i=1}^{M} (G(Y_i) - g(Y_i))^2}, \tag{V.12}$$

- le critère R

$$R^2 := 1 - \frac{1}{\sigma_g} \sum_{i=1}^{M} (G(Y_i) - g(Y_i))^2 \tag{V.13}$$

avec

- $\overline{g} := \frac{1}{M} \sum_{i=1}^{M} g(Y_i)$ la moyenne empirique de g sur les données de validation,
- $\sigma_g := \sum_{i=1}^{M} (G(Y_i) - \overline{g})^2$ la variance empirique de G sur les données de validation.

Le numérateur est la norme euclidienne au carré des résidus et le dénominateur mesure l'irrégularité de G.

Plus R^2 est proche de 1, plus l'approximation est précise.

- L'erreur absolue relative

$$R_A := \frac{1}{M \sqrt{\sigma_g}} \sum_{i=1}^{M} |G(Y_i) - g(Y_i)|. \tag{V.14}$$

Plus R_A est proche de 0, plus l'approximation est précise.

- L'erreur maximum relative

$$R_M := \frac{1}{\sqrt{\sigma_g}} \max_{i=1,\ldots,M} |G(Y_i) - g(Y_i)|. \tag{V.15}$$

Ce critère est moins important, car un R_M grand indique une erreur importante sur une zone locale, alors que les deux critères précédents R et R_A indiquent une erreur globale sur l'ensemble des données de validation.

Validation en pratique Si nous choisissons par exemple le premier critère d'erreur quadratique moyenne E, nous pouvons l'appliquer aux deux ensembles de données :

- nous notons E_C quand nous calculons E avec les données de construction,
- nous notons E_V quand nous calculons E avec les données de validation.

Le choix de la bonne approximation peut être alors donné par : trouver l'approximation avec des erreurs E_C et E_V du même ordre de grandeur et le plus petites possibles.

Dans le cas de données bruitées, le meilleur résultat possible est une erreur de l'ordre de la précision, c'est-à-dire de l'écart-type du bruit.

1.3 Plans d'expériences

Etant donné leur importance pour le choix des données de construction (hormis les méthodes d'interpolation), et de validation, nous présentons dans la suite la méthodologie des plans d'expériences, avant d'introduire les grandes classes de méthodes d'approximation.

1.3.1 Introduction

Les plans d'expériences sont utilisés dans de nombreux contextes car leur objectif est d'**apporter le plus d'information possible au meilleur coût**, en planifiant et le nombre et l'emplacement des points qu'on doit évaluer.

Pour une vue exhaustive, le lecteur peut se référer à [DFG97], [Gou99], [Gou96].

1.3.2 Applications

Les plans d'expérience sont utilisés dans tous les domaines de l'ingénieur (agriculture, chimie, biologie, électronique,...) et dans les contextes suivants.

Méthode des surfaces de réponse Nous considérons un modèle, noté G, issu de l'étude d'un système ou d'un phénomène, et décrit

- Par des expérimentations réelles :

  Souvent les expériences sont extrèmement coûteuses, et il apparaît nécessaire d'utiliser les plans d'expériences pour réduire leur nombre.

  Historiquement, les plans d'expérience se sont développés dans ce domaine.

  En général, les expériences sont non reproductibles, et leur résultat est entaché d'erreurs de mesure. Cela entraîne des spécificités comme la répétition d'une expérience pour réduire la variabilité, l'utilisation de propriétés de symétrie.

- Ou par des simulations numériques :

  Au contraire des expérience physiques, les simulations numériques ne nécessitent pas des exigences liées aux erreurs de mesure ou au bruit (ce sont des expériences déterministes).

  La plupart des méthodes d'approximation numérique (régressions, réseaux de neurones,...) s'appuient sur la méthodologie des plans d'expériences pour réduire le nombre de points à évaluer tout en assurant une information suffisante pour obtenir une approximation de qualité.

La technique des plans d'expériences va servir alors à fournir les informations pour construire un modèle approché ou approximation de G : c'est la méthode des surfaces de réponse ou RSM (pour *Response Surface Methodology*).

Historiquement, les plans d'expérience ont beaucoup servi à créer des approximations polynomiales : le placement des points du plan est alors optimisé afin de conférer les meilleures propriétés possibles à ces surfaces de réponse.

Cependant, le recours à des modèles approchés non polynomiaux est possible : dans cette étude, nous utiliserons les réseaux de neurones (voir section 2).

Analyse de sensibilité On cherche ici à déterminer de manière quantitative les variations de la réponse du modèle par rapport aux facteurs.

Nous renvoyons à la section 5.1 page 172 du chapitre précédent pour l'utilisation des plans d'expériences dans ce domaine.

1.3.3 Notations et hypothèses

Avec la terminologie des plans d'expérience

- Les **facteurs** sont les variables du modèle :

  ils peuvent être continus, discrets ou même qualitatifs.

- Les **niveaux** sont les valeurs discrètes que prennent les facteurs dans le plan d'expériences.

Généralement chaque facteur varie dans un intervalle, et le domaine expérimental est un pavé de $\mathbb{R}^n$ (mais parfois les expériences ne sont réalisables que dans une partie du domaine d'étude).

On normalise alors les valeurs sur un intervalle $[-1, 1]$ pour travailler sans dimensions : le domaine de départ est alors l'hypercube unitaire $[-1, 1]^n$ de $\mathbb{R}^n$.

1.3.4 Principe

Un plan d'expériences est représenté par une matrice :

- les n colonnes correspondent aux facteurs,
- les p lignes correspondent aux niveaux choisis pour les facteurs.

Ainsi, les lignes de cette matrice donnent les coordonnées des point du plan d'expérience.

Le nombre d'évaluations du modèle (par expérience ou simulation) définit le coût de calcul du plan et est égal à p le nombre de lignes de la matrice.

L'utilisation d'une grille complète (ou plan complet), pour une analyse du modèle avec une discrétisation de p points par dimension (p niveaux par facteur) conduit à une matrice impliquant p^n évaluations : ce nombre croît de façon exponentielle avec la dimension, et devient impraticable en grande dimension.

C'est pourquoi l'objectif des plans d'expériences est de réduire les points d'évaluation en optimisant des critères adaptés au but de l'analyse :

- des critères statistiques (D-optimalité, entropie,...),
- des critères géométriques (distance entre les points, carrés latins,...) : on les appelle ***space filling designs***.

Nous détaillons dans la suite quelques plans utilisant des critères géométriques car ils sont adaptés à notre problématiques (voir [SLC01] pour une étude comparative).

1.3.5 Plans de distance maximin

Ces plans ont été proposés dans [JM90] pour des expériences simulées.

Un plan maximin maximise la distance minimale entre deux points : les points sont distants les uns des autres et couvrent tout le domaine d'étude.

Dans le cas de la distance euclidienne, l'idée est de remplir le domaine avec sphères dont le centre est un des points du plan, et de maximiser le rayon des sphères (les sphères peuvent sortir du domaine mais pas les points) : voir illustration sur la figure V.2.

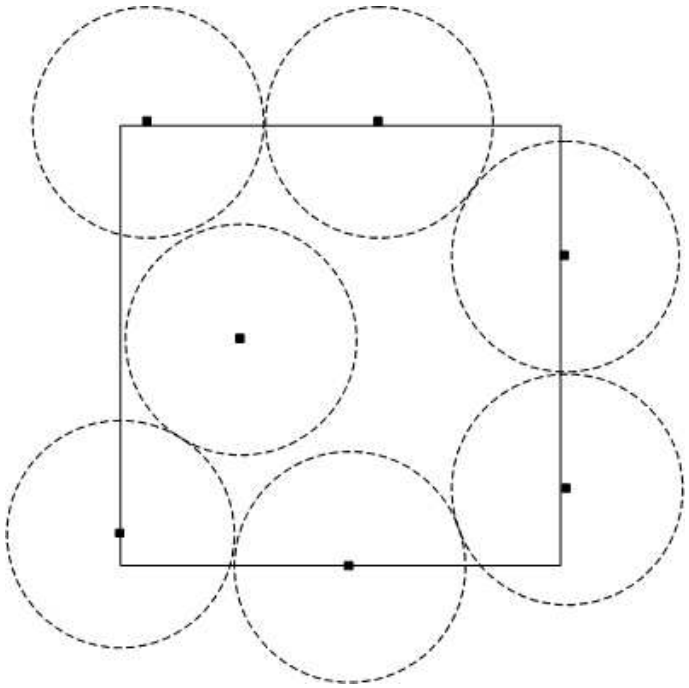


FIG. V.2 – Plan maximin pour N=7 points dans $[-1,1]^2$

L'inconvénient est que les points ont tendance à se concentrer sur les frontières du domaine, et si on effectue une projection suivant une direction, les points ne sont pas forcément bien répartis.

1.3.6 Carré latin

Le plan de type carré latin a été introduit dans [MBC79], parmi les premiers à utiliser les plans d'expériences pour des expériences déterministes. Ils ont été utilisés ensuite pour la planification et l'analyse de simulations (voir [SWMW89]).

Les carrés latins sont un cas particulier des tirages hypercubes latins qui ont été présentés en section 3.2.3 du chapitre précédent pour améliorer les tirages aléatoires simples.

Ils utilisent une méthode de stratification, afin d'estimer la distribution d'une réponse fonction de paramètres incertains suivant une distribution donnée.

Il suffit de considérer que les paramètres suivent des lois uniformes, et pour un plan de p points, l'intervalle de variation est alors partagé régulièrement en p niveaux, et la matrice de plan d'expérience d'un hypercube latin est alors de la forme :

- chaque colonne contient les p niveaux permutés aléatoirement,
- les colonnes sont permutées aléatoirement.

Ces propriétés assurent que chaque niveau est représenté une fois par paramètre, les points sont bien répartis et on évite les inconvénients du plan maximin : voir une illustration sur la figure V.3.

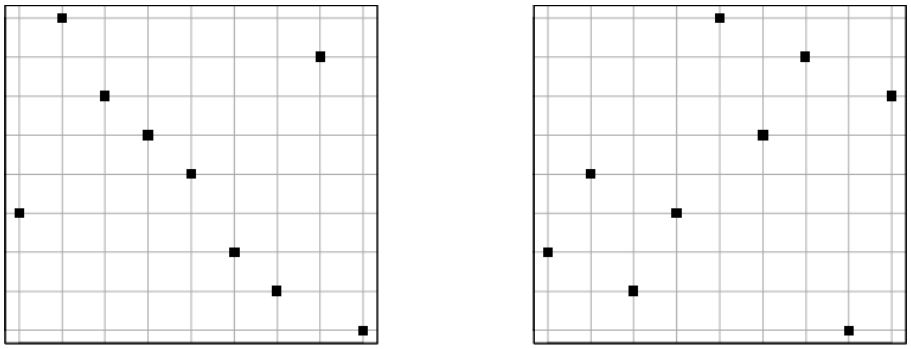


FIG. V.3 – Carré latin pour deux paramètres et 9 expériences

1.3.7 Autres plans

Nous signalons l'existence d'autres plans adaptés pour générer des données de construction d'approximations :

- les plans orthogonaux,
- les plans à discrépance réduite,
- les plans uniformes.

1.3.8 Conclusion

Notre objectif est une analyse d'incertitude, effectuée avec des paramètres définis par des lois de probabilités.

Nous envisageons d'utiliser un plan d'expérience adapté à la construction d'une approximation pour un modèle dont on ne connaît pas la régularité, afin de résoudre les trois problèmes (P_i) de la section 2.2 du chapitre précédent (voir page 156) :

- (P_1) probabilités de défaillance,
- (P_2) points de conception,
- (P_3) paramètres influents.

Le tirage hypercube latin semble bien adapté grâce aux propriétés suivantes :

- on contôle le nombre de points,
- une bonne distribution suivant chaque direction est assurée, ce qui favorise la détection d'irrégularités et la détection des paramètres influents : par rapport à d'autres plans d'expériences, un nombre minimum de points est placé à l'intérieur du domaine d'étude.
- l'échantillon respecte la loi de probabilité des paramètres, mais répartit les coordonnées dans toutes les zones d'équiprobabilité, ce qui permet d'obtenir des évaluations, c'est-à-dire l'information, dans des zones contribuant à des faibles probabilités de défaillance.

1.4 Approximations classiques

1.4.1 Approximation polynomiale

Les polynômes multivariés sont couramment utilisés comme modèle approché, souvent à l'ordre un ou deux.

Dans l'approximation polynomiale, on utilise la méthode des moindres carrés pour l'ajustement.

Approximation linéaire Dans cette approche, on cherche g sous la forme d'un polynôme du premier degré

$$g(X, B) = b_0 + B^T\ X \tag{V.16}$$

avec $B = (b_1, \ldots, b_n) \in \mathbb{R}^n$ et $b_0 \in \mathbb{R}$ les coefficients du modèle à estimer à partir des des données simulées.

Approximation quadratique Afin de rendre compte de comportements non linéaires, on cherche g sous la forme d'un polynôme de degré deux

$$g(X, B) = b_0 + B^T\ X + X^T\ C\ X \tag{V.17}$$

avec $C \in \mathbb{R}^{n \times n}$.

Cette approximation, dite quadratique, qui prend mieux en compte la géométrie de la fonction de modélisation G, est souvent utilisée dans la pratique.

Les plans classiquement utilisés pour un modèle polynomial de degré deux sont les plans composites centrés ou les plans de Box-Behnken (voir [Gou99]).

Le développement s'écrit en fonction des monômes et linéairement par rapport à B

$$g(X, B) = b_0 + \sum_{i=1}^{n} b_i X_i + \sum_{i=1}^{n} b_{ii} X_i^2 + \sum_{i=1}^{n} \sum_{j<i} b_{ij} X_i X_j. \tag{V.18}$$

Approximation d'ordre supérieur Si on choisit un polynôme de degré supérieur, on écrit alors sous une forme plus générale

$$g(X, B) = B^T \; r(X) \tag{V.19}$$

avec

- $B \in \mathbb{R}^p$ les coefficients du développement (généralisation du développement quadratique selon les monômes précédent) à calculer,
- $r(X) = (1, r_1(X), \ldots, r_{p-1}(X))^T \in \mathbb{R}^p$ le vecteur des monômes.

Généralement, on ne prend pas en compte tous les monômes, et on tronque plus ou moins le développement complet : sinon le calcul devient beaucoup plus complexe et coûteux (le nombre de données pour calculer tous les coefficents devient très important), avec des comportements non désirées entre les données (instabilités numériques).

Ajustement Pour calculer les coefficients B qui déterminent $g(X, B) = B^T \; r(X)$, la méthode la plus utilisée est celle des moindre carrés, dits linéaires, car g est linéaire par rapport à B (et non par rapport à X).

Cette méthode est très simple à mettre en oeuvre car les coefficients b sont directement donnés par la résolution du système linéaire

$$R^T R \; B = R^T \; y \tag{V.20}$$

avec

- $y := (G(X_1), \ldots, G(X_N))^T \in \mathbb{R}^N$ le vecteur dit des observations pour G (on prendra $N > p$),
- $R := (r_j(X_i))_{ij} \in \mathbb{R}^{N \times p}$ la matrice des observations pour les monômes.

Conclusion Les approximations linéaires ou quadratiques sont souvent valides même pour approcher des fonctions complexes mais plutôt au niveau local, au voisinage d'un point.

Elles supposent une dépendance entre les paramètres d'ordre un ou deux pour la fonction de modélisation G, or en pratique pour un modèle décrivant un système complexe avec de nombreux paramètres d'entrée, ce n'est pas le cas, et ces approximations ne peuvent pas rendre compte de toutes les tendances (même pas approcher toutes les données).

Augmenter le degré des polynômes de l'approximation donne plus de degrés de libertés, et répond à cette attente, mais au prix d'une plus grande complexité et d'instabilités de l'approximation, partiellement résolues par une méthode de régularisation (voir section 1.5.2 page 198).

1.4.2 Interpolation polynomiale

Dans l'interpolation on fait coïncider sur les données la valeur de la fonction G et de l'approximation g.

En dimension un, il existe un unique polynôme de degré inférieur ou égal à $N-1$ passant par N données différentes.

La convergence globale mesurée par $\|G-g\|$ n'est pas toujours assurée, et on rencontre par exemple en dimension un le phénomène de Runge : des oscillations de plus en plus grandes apparaissent quand le degré du polynôme augmente.

Splines L'interpolation spline évite les problèmes d'oscillations précédents, car elle permet d'utiliser sur chaque segment un polynôme de degré plus faible.

Une courbe spline en dimension un est une famille de polynômes de dimension donnée définis sur des segments $[x_i, x_{i+1}[$ formant une partition de l'intervalle de définition et de courbure minimale : les polynômes sont raccordés aux points x_i ainsi que leur dérivées jusqu'à un certain ordre pour obtenir une approximation régulière.

Grande dimension Ces techniques s'étendent aux dimensions supérieures (avec la méthode MARS pour *Multivariate Adaptive Regression Splines* par exemple).

Cependant les méthodes d'interpolation classiques sont basées sur un nombre de points trop important pour être utilisées en grande dimension, mais ce problème est en partie résolu par la technique des *sparse grids* (voir section 3) que nous mettrons en oeuvre dans la suite.

Eléments finis Cette méthode très connue est une méthode d'interpolation : elle permet de résoudre des équations aux dérivées partielles de manière numérique. On cherche une solution approchée $g \in \mathcal{V}_h$ de la solution $G \in \mathcal{V}$ sur un domaine compact avec des conditions au bord pour assurer l'existence et l'unicité de la solution.

Le domaine est discrétisé par un maillage $\{X_1, \ldots, X_N\}$ dont les sommets sont les noeuds : souvent on travaille en dimension deux et les éléments sont des triangles, mais ils peuvent être aussi des polyèdres.

On associe au maillage des fonctions de base w_i pour définir $\mathcal{V}_h$:

- Généralement on définit une fonction de base w_i par noeud X_i, qui vaut un au noeud X_i et zéro ailleurs

$$\forall i, j = 1, \ldots, N \quad w_i(X_j) = \delta_{ij}. \tag{V.21}$$

- Les fonctions de base sont par exemple des fonctions continues et affines par morceaux sur les éléments du maillage.

La dimension de $\mathcal{V}_h$ est alors N, et dans ce cas g est déterminée par les valeurs de G aux noeuds du maillage avec

$$g(X) = \sum_{i=1}^{N} G(X_i)\ w_i(X). \tag{V.22}$$

La méthode des éléments finis se généralise à une dimension quelconque, mais en pratique on résoud des problèmes de dimension inférieure ou égale à trois : si elle permet de disposer de théorèmes de convergence et de contrôle d'erreur, son inconvénient est le coût de calcul, induit par le nombre important de noeuds du maillage qu'il faut définir.

Remarque En pratique, dans le cas de données expérimentales entachées de bruit, l'interpolation n'est pas adaptée : il est clair que l'utilisateur qui travaille avec 100 points presque alignés préfère la droite passant au mieux entre les points (moindres carrés) que le polynôme de degré 99 passant par tous les points (interpolation).

1.5 Approches complémentaires

1.5.1 Réduction de modèle

Pour des problèmes dont la dimension n est très grande (de l'ordre du millier de variables), aucune méthode d'approximation directe n'est efficace à cause du coût de calcul.

De plus, l'expérience montre qu'un nombre réduit de ces variables est susceptible de contenir l'essentiel des variations de la réponse G du modèle : si le système modélisé est d'origine physique, seules quelques grandes forces expliquent en général le phénomène observé.

Il apparaît alors utile d'utiliser des techniques dites de **réduction de modèle**, qui consistent à **négliger certaines variables pour réduire la dimension de l'espace de travail** : dans ce cas, on construit g comme une fonction de m paramètres, avec $m < n$, parmi les paramètres initiaux.

La réduction de modèle peut s'appliquer

- à partir de considérations physiques (avis d'expert),
- après une étude préalable (comme une analyse de sensibilité, voir section 5.1 du chapitre précédent page 172), qui permet d'identifier les paramètres importants et négligeables,
- à partir de méthodes de projection (voir [ASS01]) telles que
  - les méthodes basées sur une décomposition SVD (pour *Singular Value Decomposition*) afin de trouver une base réduite : la méthode POD (pour *Proper Orthogonal Decomposition*), l'analyse en composantes principales,
  - la méthode des sous-espaces de Krylov.

Cette approche est essentielle car dans le cas de modèles en grande dimension, confronté à une limite en coût de calcul, l'utilisateur peut choisir une approximation simple (par exemple un polynôme du second degré) avec peu de degrés de liberté qui génère peu de calculs de construction, mais qui aura une faible capacité de généralisation.

Il vaut mieux alors consacrer une partie du coût de calcul à effectuer une réduction de modèle : la construction d'une approximation plus sophistiquée (par exemple un

réseau de neurones) portant uniquement sur les variables les plus importantes sera possible car moins coûteuse, et la capacité de généralisation sur le modèle initial sera plus importante.

1.5.2 Régularisation

Dans l'approche par minimisation de l'erreur, soit $J(B)$ la fonction coût à minimiser pour trouver les coefficients B.

Par exemple dans le cadre des moindres carrés pour un polynôme

$$J(B) = \sum_{i=1}^{N} [G(X_i) \; - \; b^T \; r(X_i)]^2. \tag{V.23}$$

L'ajustement n'est pas l'objectif principal de la construction de l'approximation, c'est la qualité de prédiction, or la première propriété n'implique pas la deuxième : l'ajustement peut être parfait ($J(b) = 0$) et la généralisation médiocre.

Par exemple en ce qui concerne une approximation polynomiale de degré p, le nombre de termes croît très rapidement avec la dimension (un développement complet comporte C_{n+p}^{n} termes).

Or un nombre grand de coefficients à estimer conduit souvent à un problème mal posé et à des instabilités numériques : des oscillations non désirées apparaissent quand le degré du polynôme s'accroît (phénomène de sur-ajustement, voir section 1.2.5 page 186).

La régularisation est une méthode classique pour rendre un problème bien posé : le principe est d'ajouter des contraintes supplémentaires afin de rendre la solution unique, ou réduire l'ensemble des solutions en écartant les solutions non désirées, comme celles qui présentent des instabilités numériques dues au sur-ajustement.

Plusieurs techniques existent, nous les utiliserons notamment dans le cadre de l'approximation par réseaux de neurones (voir section 2.3.5 page 215 pour la mise en oeuvre).

Régularisation de Tikhonov Une solution classique est la régularisation de Tikhonov : on incorpore à la fonction coût un terme supplémentaire, dont la minimisation apporte les propriétés désirées pour la solution.

Le problème régularisé s'écrit alors par exemple

$$J_K(B) = J(B) + K \; \|B\|^2 \tag{V.24}$$

avec $K \in \mathbb{R}^+$ une constante.

Ainsi, suivant la valeur plus ou moins grande donnée à K, la minimisation de J_K effectue un compromis entre la minimisation des deux termes :

- faire correspondre les valeurs du modèle et de l'approximation sur les données en minimisant $J(B)$,
- et pénaliser les grandes valeurs des coefficients B pour éviter les instabilités : garder une norme de B petite permet de réduire les oscillations.

1.5.3 Krigeage et modèle composite

Introduction Le krigeage, ou *kriging*, est une méthode d'estimation d'un modèle défini par une fonction aléatoire : il prend en compte la variabilité des phénomènes étudiés dans le résultat de l'approximation, et contrairement aux méthodes précédentes, permet une quantification des erreurs. C'est aussi une méthode optimale d'estimation, au sens statistique du terme.

Cette méthode entre dans le cadre de la modélisation où les données sont issues de mesures bruitées, mais nous verrons ci-dessous qu'elle aussi utilisée dans le cadre déterministe pour approcher une fonction de modélisation.

Cette méthode est dite aussi "géostatistique", et on rencontre le terme de "méthode d'interpolation de Gauss-Markov" en statistique.

Cette technique a été introduite dans [Mat63], qui a utilisé le formalisme des variables aléatoires pour l'estimation de phénomènes naturels : les termes "géostatistique" et "krigeage" proviennent des travaux d'un ingénieur, D.G. Krige dans les années 50, pour des problèmes d'exploitation minière (il fallait déterminer au mieux la distribution spatiale de minerais à partir d'un échantillon de prélèvement du sous-sol). Cette technique a été développée à la même époque dans le domaine de la météorologie dans [Gan65] et un peu plus tard dans le domaine de l'océanographie dans [BDF76].

De manière naturelle, le krigeage a été largement utilisé dans le domaine de la géologie, limité à des problèmes en dimension trois. La méthode se généralise à une dimension quelconque, mais son apparition dans d'autres domaines est récente : voir par exemple [BS92] et [SWMW89] pour l'application à des simulations numériques.

Le krigeage est actuellement appliqué par exemple pour des analyses de risques en ingénierie pétrolière (voir [Sch06]) et de manière plus générale pour des estimations, surtout dans les domaines de l'environnement (planification des mines et des gisements, cartographie météorologique, estimation de la biomasse et de sa localisation pour la pêche, mais aussi analyse et caractérisation d'images,...).

Nous présentons ici le principe du krigeage, le lecteur intéressé par un exposé plus complet pourra consulter [DJ06].

Hypothèses Soit $\Omega \subset \mathbb{R}^n$ l'espace de travail (généralement $n = 2$ ou 3), nous considérons ici un modèle Z à prédire, mais contrairement à ce qui précède avec la fonction de modélisation G, en tout point $X \in \Omega$, $Z(X)$ est ici défini comme une variable aléatoire à valeurs dans $\mathbb{R}$.

Nous appelons alors fonction aléatoire Z l'ensemble des variables aléatoires $\{Z(X),\ X \in \Omega$. On dit que les variables aléatoires sont régionalisées, c'est-à-dire qu'elles dépendent de leur localisation X.

Le krigeage cherche alors en tout point X à estimer $Z(X)$ par une valeur notée $\widehat{Z}(X)$, et $\widehat{Z}$ caractérise Z globalement.

On définit l'erreur d'estimation e par

$$e(X) := \widehat{Z}(X) - Z(X). \tag{V.25}$$

La variance de l'estimation $V(e)$ s'écrit alors

$$V[e(X)] = V[Z(X)] + V[\widehat{Z}(X)] - 2\ cov[Z(X), \widehat{Z}(X)]. \tag{V.26}$$

Nous disposons de N données (ou observations) $\{Z(X_i),\ i = 1, \ldots, N\}$ comme information pour prédire les valeurs de Z : généralement une seule observation en chacun des N points.

Variogramme La différence entre les valeurs prises par Z en deux points X et $X + H$ est la variable aléatoire $Z(X) - Z(X + H)$: on veut alors que sa variance soit plus petite quand les points sont proches, c'est-à-dire $\|H\|$ petit, que quand ils sont éloignés (car les valeurs se ressemblent plus en moyenne).

On définit alors le "variogramme" γ, qui est une statistique d'ordre deux décrivant la continuité spatiale du phénomène

$$\gamma[Z(X_1), Z(X_2)] := \frac{1}{2}\ (V[Z(X_1) - Z(X_2)]). \tag{V.27}$$

Si l'on considère N localisations différentes $X_1, \ldots, X_N$, la meilleure description des N variables aléatoires $Z(X_1), \ldots, Z(X_N)$ est de calculer la densité de probabilité jointe : ce n'est pas possible car on ne dispose pas d'une information suffisante.

On cherchera à estimer plus simplement les deux premiers moments (moyenne, variance, covariance) des $Z(X_i)$ prises deux à deux, et les hypothèses suivantes, qui supposent une certaine régularité, ou homogénéité, sont alors nécessaires.

On suppose la "stationnarité du second ordre" (on peut formuler des hypothèses moins restrictives avec des résultats différents), c'est-à-dire

- L'espérance de Z est constante sur le domaine

$$\forall X \in \Omega \quad E[Z(X)] = m. \tag{V.28}$$

- La covariance de Z est constante par translation

$$\forall H \in \mathbb{R}^n\ \forall X \in \Omega \quad C(H) := cov[Z(X), Z(X + H)]. \tag{V.29}$$

On peut écrire donc que la variance de Z est constante sur le domaine avec

$$\forall X \in \Omega \quad V[Z(X)] = C(0). \tag{V.30}$$

Cela implique que le variogramme aussi ne dépend que de l'écart H entre les points, $\gamma : \mathbb{R}^n \longrightarrow \mathbb{R}$, et se simplifie avec l'hypothèse de stationnarité

$$\begin{aligned} \forall H \in \mathbb{R}^n\ \forall X \in \Omega \quad \gamma(H) &= \tfrac{1}{2}\ V[Z(X + H) - Z(X)] \\ &= \tfrac{1}{2}\ E[(Z(X + H) - Z(X))^2] \\ &= C(0) - C(H). \end{aligned} \tag{V.31}$$

Estimation du variogramme Chaque phénomène possède un variogramme qui lui est propre, il est donc nécessaire de l'estimer à partir des données.

La méthode est la suivante :

- Soit $H := X_i - X_j$ l'écart entre deux points des données, on détermine toutes les valeurs $H_1, \ldots, H_m$ possibles : en pratique, on s'accorde une tolérance sur H afin d'avoir suffisamment de paires d'observations pour chaque H.
- On calcule une estimation du variogramme pour les valeur $H_1, \ldots, H_m$, c'est le variogramme dit "expérimental"

$$\gamma(H_i) = \frac{1}{2}\,\frac{1}{n_i}\,\sum_{i=1}^{n_i}(Z(X_i) - Z(\widetilde{X}_i))^2 \tag{V.32}$$

  avec n_i le nombre de paires de points $\{X_i, \widetilde{X}_i\}$ qui vérifient $H_i := X_i - \widetilde{X}_i$.

  On obtient donc une série de points expérimentaux.
- On ajuste alors une fonction analytique pour obtenir l'estimation du variogramme pour H quelconque.

  Il existe différents modèles de variogrammes utilisés en pratique (sphérique, cubique, gaussien, ...).

La détermination du modèle du variogramme influe sur l'approximation finale : pour une illustration en dimension un, voir la figure V.4.

Principe L'idée à la base du krigeage est la suivante : deux observations situées l'une près de l'autre devraient, en moyenne, se ressembler davantage que deux observations éloignées.

Toutes les méthodes d'estimation ou d'approximation reposent sur ce concept plus ou moins défini, plus ou moins implicite de "continuité" entre les données : avec le krigeage, on cherche à quantifier cette continuité préalablement à tout calcul, grâce au variogramme.

L'avantage du krigeage est qu'il permet de calculer en tout point X une estimation $\widehat{Z}(X)$ et la variance associée $V[e(X)]$, qui mesure l'erreur de prédiction.

Krigeage ordinaire Le krigeage dit "ordinaire" est plus fréquemment utilisé, et ne suppose pas la moyenne m du phénomène connue.

L'estimation notée $\widehat{Z}$ de Z en un point X_0 non observé, utilise une combinaison linéaire des données

$$\widehat{Z}(X_0) := \sum_{i=1}^{N} w_i^0\ Z(X_i) \tag{V.33}$$

$$= W_0^T\ Z \tag{V.34}$$

avec

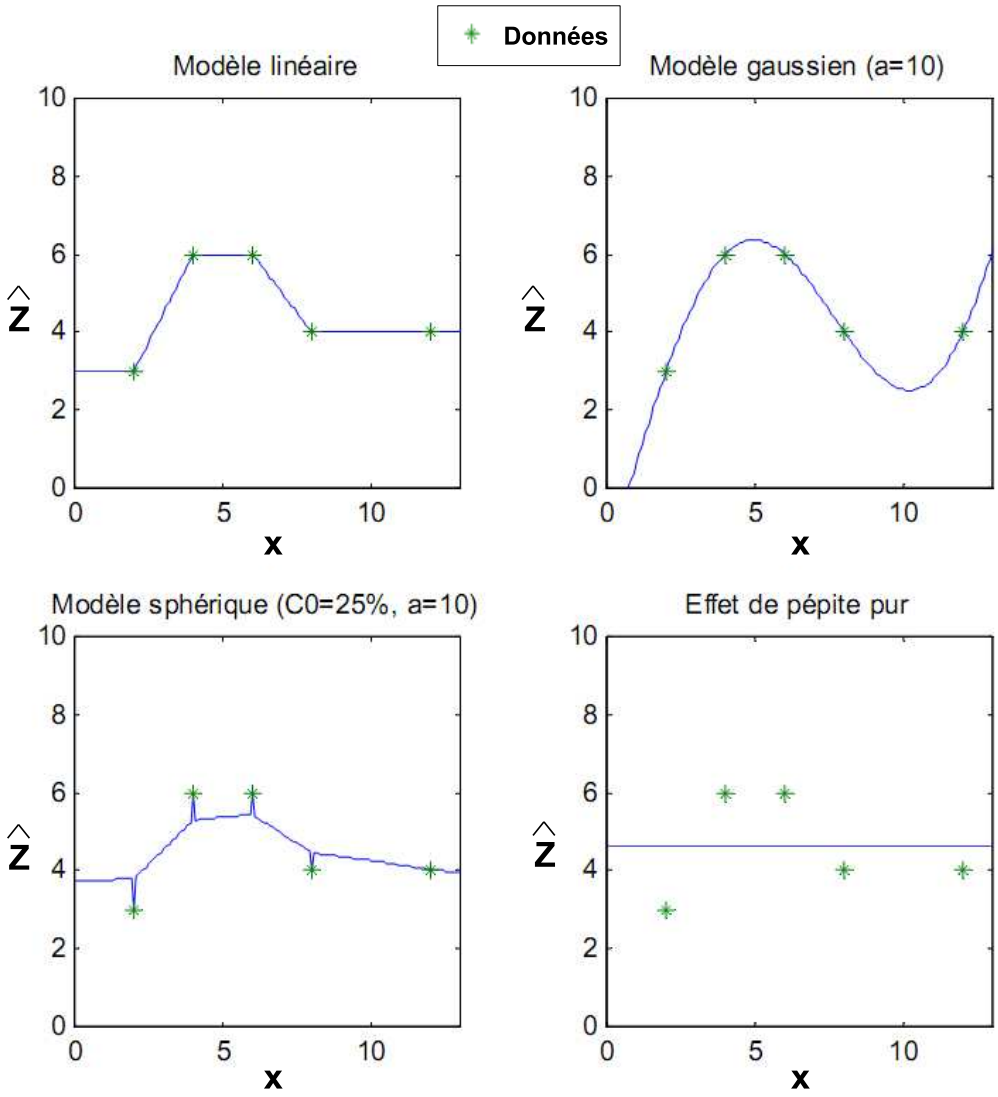


FIG. V.4 – Exemple de krigeage 1D avec différents modèles de variogramme

- $W_0^T := (w_1^0, \ldots, w_N^0)$ les coefficients ou poids à calculer, qui dépendent du point X_0,
- $Z^T := (Z(X_1), \ldots, Z(X_N))$ les données.

Pour que cet estimateur soit sans biais, on impose

$$\sum_{i=1}^{N} w_i^0 = 1. \tag{V.35}$$

Le krigeage ordinaire est un estimateur linéaire prenant en compte la covariance spatiale des données : pour déterminer les poids W_0, on va minimiser la variance d'estimation $V(e(X_0))$.

En substituant $\widehat{Z}$ par son expression (V.33) comme combinaison linéaire des don-

nées, on obtient

$$V[e(X_0)] = V[Z(X_0)] + \sum_{i=1}^{N}\sum_{j=1}^{N} w_i^0 w_j^0 \; cov[Z(X_i), Z(X_j)] - 2\sum_{i=1}^{N} w_i^0 w_j^0 \; cov[Z(X_0), Z(X_i)]. \tag{V.36}$$

Les coefficients W_0 s'obtiennent en résolvant le problème de minimisation d'une fonction quadratique sous contrainte d'égalité

$$\begin{cases} \min\limits_{W_0 \in \mathbb{R}^N} J(W_0) := V[e(X_0)] \\ \\ \sum_{i=1}^{N} w_i^0 = 1. \end{cases} \tag{V.37}$$

L'expression $V[e(X_0)]$ peut s'exprimer en fonction du variogramme $\gamma(H)$, et après avoir écrit le Lagrangien, la condition d'optimalité s'écrit comme un système linéaire à résoudre qui s'écrit en fonction du variogrammme

$$\begin{cases} \forall i = 1, \ldots, N \quad \sum_{j=1}^{N} w_i^0 \; \gamma[Z(X_i), Z(X_j)] + \mu = \gamma[Z(X_0), Z(X_i)] \\ \sum_{i=1}^{N} w_i^0 = 1 \end{cases}$$

avec

- μ le multiplicateur de Lagrange associé à la contrainte d'égalité.

Une fois les poids w_i^0 calculés, nous pouvons écrire une estimation $\widehat{Z}(X_0)$ de la valeur de $Z(X_0)$ avec la relation (V.33), puis une estimation de l'erreur $V[e(X_0)]$, qui s'écrit en fonction des valeurs trouvées ; on peut aussi donner le résultat avec un intervalle de confiance.

Utilisation dans le cadre déterministe Le krigeage, qui à l'origine permet l'approximation de processus spatiaux avec des mesures bruitées, est aussi utilisé pour traiter des problèmes d'approximation où les données sont simulées, donc déterministes.

Le principe a été proposé dans [SWMW89], et est utilisé sous les termes de "*kriging metamodel*" ou "modèle composite" dans [Sch06] par exemple.

Soit $G(X)$, la fonction de modélisation déterministe, on décompose ainsi l'approximation

$$\mathbf{g}(X) = f(X) \; + \; \mathbf{Z}(X) \tag{V.38}$$

avec

- $f(X)$ une approximation globale, souvent un polynôme d'ordre 1 ou 2,
- $\mathbf{Z}(X)$ est défini en tout point X comme la réalisation d'un processus aléatoire d'espérance nulle et de variance constante σ^2.

L'approximation $\mathbf{g}$ est ainsi composée de deux termes

- f est la partie déterministe, qui modélise les tendances moyennes de la fonction à approcher.

  f est déterminée avec une régression par moindres carrés sur les données.

- $\mathbf{Z}$ est un terme correcteur, qui permet des déviations locales de l'approximation, de façon à ce que $\mathbf{Z}$ interpole les résidus, et donc $\mathbf{g}$ interpole les données.

$\mathbf{Z}$ présente une structure de corrélation entre les données, la covariance est alors donnée par

$$cov[\mathbf{Z}(X_i), \mathbf{Z}(X_j)] = \sigma^2 \; c[\mathbf{Z}(X_i), \mathbf{Z}(X_j)] \tag{V.39}$$

avec c la corrélation, qui est choisie parmi une classe de fonctions de corrélation usuelles.

Ce choix quantifie la manière plus ou moins "lisse" (avec des variations rapides ou non) avec laquelle $\mathbf{g}$ va passer d'un point à un autre et donc interpoler les données. En simulation numérique, la plus utilisée est la corrélation gaussienne proposée dans [SWMW89].

Le terme correcteur $\mathbf{Z}$ est déterminé par krigeage : soient les données $\{X_i,\ i = 1, \ldots, N\}$, et $Z := (\widehat{Z}(X_1), \ldots, \widehat{Z}(X_N))^T = (G(X_1) - f(X_1), \ldots, G(X_N) - f(X_N))^T$ le vecteur des résidus.

En un point X_0 quelconque, on estime le résidu $G(X_0) - f(X_0)$ par une combinaison linéaire des observations

$$\widehat{Z}(X_0) = \lambda_0^T \; Z \tag{V.40}$$

avec $\lambda_0^T := ({\lambda_1^0}^T, \ldots, {\lambda_N^0}^T)^T$ les coefficients à calculer.

Après minimisation de la variance de l'erreur en X_0 et simplification par σ^2, on obtient

$$\widehat{Z}(X_0) = r(X_0)^T \; R^{-1} \; Z \tag{V.41}$$

avec

- $r(X_0) := (c(X_0, X_1), \ldots, c(X_0, X_N))^T$ le vecteur de corrélation entre le point à estimer et les données,
- $R := \sigma^2 \; (c(X_i, X_j))_{i,j \in \{1,\ldots,N\}}$ la matrice de corrélation entre les données.

Finalement, l'approximation par modèle composite en X_0 s'écrit

$$\widehat{g}(X_0) = f(X_0) \; + \; \widehat{Z}(X_0) \; \simeq G(X_0). \tag{V.42}$$

Cette méthode présente les avantages

- d'être souple : on peut choisir des fonctions de corrélation qui donnent des approximations différentes,
- d'estimer l'erreur hors des données simulées.

Mais elle peut être complexe à mettre en oeuvre et coûteuse, car par exemple dans le cas de la corrélation gaussienne, déterminer les paramètres de la corrélation par maximum de vraisemblance nécessite la résolution d'un problème d'optimisation coûteux.

2 Réseaux neuronaux

2.1 Principe

2.1.1 Introduction

Un **réseau de neurones** est une fonction obtenue à partir d'une combinaison de fonctions de base, appelées neurones car leur fonctionnement présente des similitudes avec les éléments à la base du système nerveux.

Comme dans le cerveau, le réseau est déterminé par les connexions entre ses neurones, qui transmettent l'information.

Les applications sont nombreuses

- l'approximation de fonctions (pour l'optimisation, le contrôle,...),
- la classification (reconnaissance de formes,systèmes radar,...),
- la fouille de données ou *data mining* (applications financières, diagnostic médical,...).

Le lecteur trouvera un exposé complet sur le sujet dans [Hay99] ou [DMS+02].

2.1.2 Neurone formel

Un **neurone formel** est une fonction de la forme

$$\begin{array}{rccc} N: & \mathbb{R}^d & \longrightarrow & \mathbb{R} \\ & X=(x_1,\ldots,x_d) & \longmapsto & N(x)=f\left(\sum_{i=1}^{d} b_i\ X_i + b_0\right) \end{array} \tag{V.43}$$

avec

- $b=(b_0,b_1,\ldots,b_d)$ un vecteur de coefficients appelés **poids**,
- $f:\mathbb{R}\longrightarrow\mathbb{R}$ un **fonction** dite **d'activation** ou de transfert.

On peut effectuer une analogie avec le neurone biologique (voir figure V.5)

- les variables d'entrée $(X_1,\ldots,X_d)$ correspondent aux signaux électriques qui transmettent l'information en provenance d'autres neurones biologiques,
- les poids b_i sont appliqués aux entrées, et leur sommation pondérée constitue l'influx nerveux reçu par le neurone biologique à travers les connexions synaptiques(un poids positif aura donc un effet excitateur, et négatif aura un effet inhibiteur),
- le neurone biologique est activé quand un certain seuil d'influx nerveux est atteint : il transmet alors à son tour une certaine quantité d'information aux neurones avec lesquels il est relié.

La somme pondérée des entrées constitue la partie linéaire du neurone, en choisissant une fonction d'activation non linéaire, le neurone formel est une fonction non linéaire bornée.

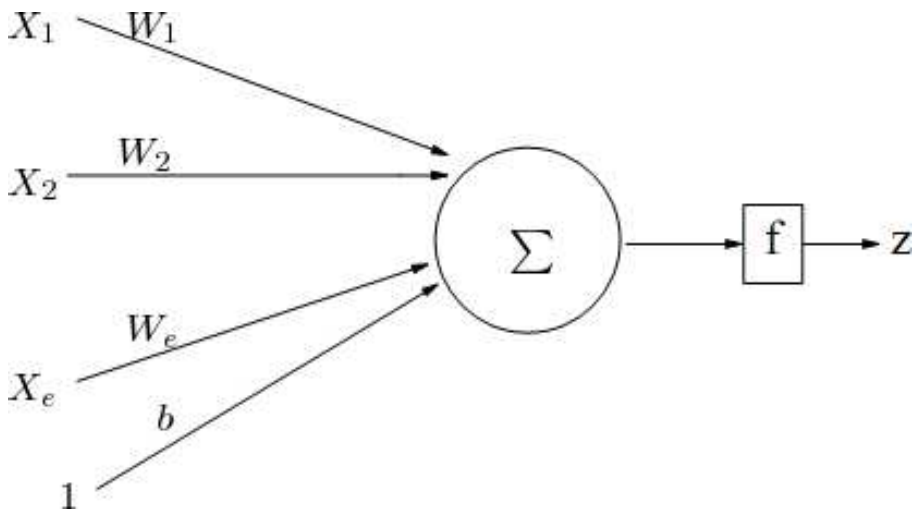


FIG. V.5 – Principe du neurone formel

2.1.3 Fonction d'activation sigmoïde

Généralement, les fonctions d'activation sont à valeur dans $[0,1]$ ou $[-1,1]$ et selon leur nature, elles limitent plus ou moins la sortie du neurone.

Une classe de fonctions très utilisée en pratique car d'usage recommandé est celle des **fonctions sigmoïdes**, c'est-à-dire des fonctions avec une forme de "s".

Plusieurs formulations existent, (par exemple la tangente hyperbolique ou Arctangente), nous prenons la suivante

$$\begin{array}{rccc} f_\alpha : & \mathbb{R} & \longrightarrow & [0,1] \\ & x & \longmapsto & f_\alpha(x) = \frac{1-e^{-\alpha x}}{1+e^{-\alpha x}} \end{array} \tag{V.44}$$

avec $\alpha \in \mathbb{R}^+$.

Comme on le voit sur la figure V.6, la valeur du paramètre α conditionne la pente de la sigmoïde et donc son effet sur l'entrée

- Lorsque α est proche de zéro, la sigmoïde se rapproche d'une fonction linéaire,
- lorsque α est grand, la sigmoïde se rapproche d'une fonction de type signe, avec deux valeurs de seuil : $f_\alpha(x) = -1$ pour les valeurs de x les plus petites et $f_\alpha(x) = 1$ pour les valeurs de x les plus grandes.

  De là vient l'analogie avec le seuil à partir duquel le neurone biologique transmet l'information.

Remarque On trouve aussi des neurones dont le fonctionnement diffère du modèle de neurone présenté précédemment : il n'y pas de somme pondérée des entrées, les coefficients qui déterminent la fonction du neurone interviennent directement dans sa définition.

Les fonction utilisées ici peuvent être

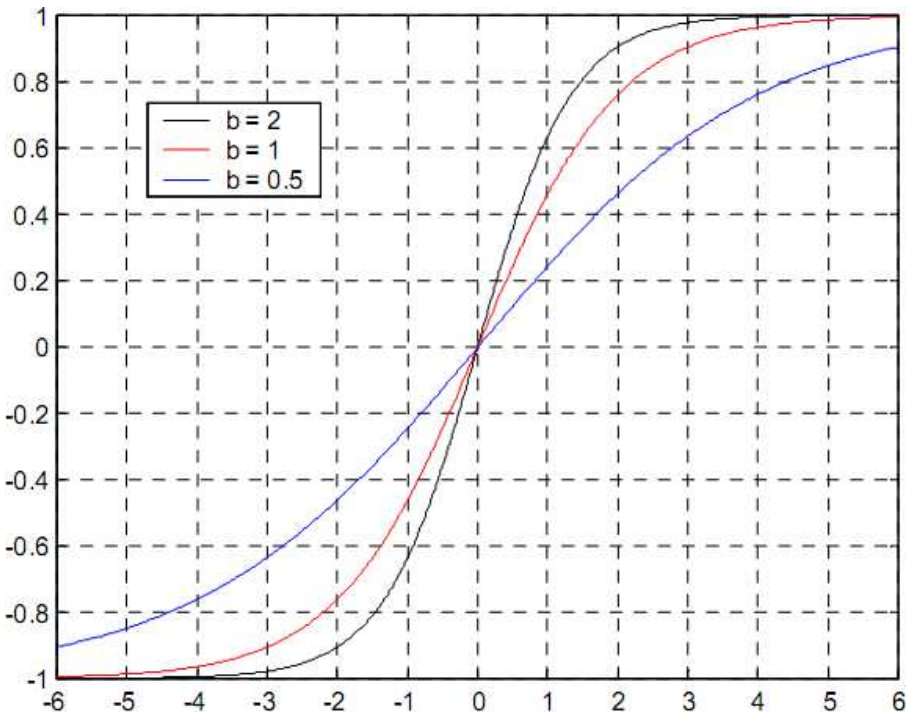


FIG. V.6 – Fonction sigmoïde pour différentes valeurs du paramètre α

- des fonctions radiales, dites RBF (pour *Radial Basis Functions*) : leur sortie ne dépend que de la distance à un point c appelé centre

$$f(x) = f(\|x\|), \tag{V.45}$$

- ou des ondelettes.

La sortie d'un neurone RBF en choisissant une fonction gaussienne est par exemple

$$N(x) = \exp\left(-\frac{\sum_{i=1}^{d}(x_i - b_i)^2}{2\ b_0^2}\right) \tag{V.46}$$

avec $b = (b_0, b_1, \ldots, b_d)$ le vecteur des coefficients, avec

- b_0 l'écart-type de la gaussienne,
- $(b_1, \ldots, b_d)$ les coordonnées du centre de la gaussienne.

La différence avec le type de neurone précédent est que les fonctions RBF ou ondelettes utilisées ici ont des non-linéarités locales, qui tendent vers zéro dans toutes les directions. Leur zone d'influence est donc limitée dans l'espace des entrées, contrairement aux neurones avec fonction d'activation sigmoïde.

2.1.4 Types de réseau de neurones

Le réseau de neurones est constitué par un ensemble de neurones reliés par des connexions décrites précédemment : on peut ainsi construire une fonction de $\mathbb{R}^n$ à valeurs dans $\mathbb{R}^p$ comme combinaison de fonctions de base (les neurones formels).

La **structure** du réseau, ou **architecture**, peut être décrite par un graphe représentant les entrées, les connexions, et les sorties.

Différentes architectures ont été définies, associant généralement plusieurs couches de neurones entre elles. Elles se différencient par la manière dont les neurones sont disposés et sont connectés entre eux.

- Un **réseau bouclé**, ou récurrent ou dynamique, présente un graphe avec des cycles (des chemins composés de connexions qui reviennent au point de départ).

  La notion de temps est ici prise en compte en affectant un coefficient appelé retard aux connexions : tout cycle du graphe du réseau doit avoir un retard non nul pour que le réseau définisse une fonction.

  Ce type de réseau est utilisé pour identifier des systèmes dynamiques non linéaires.

- Un **réseau non bouclé**, ou statique, présente un graphe sans cycle : l'information se propage de la couche d'entrée vers la couche de sortie sans retour en arrière.

  Généralement, les neurones sont alors organisés par couches successives et les connexions s'établissent entre les couches.

  Ce type de réseau est utilisé pour appprocher des fonctions non linéaires.

  Le perceptron multicouches est sans doute le plus simple et aussi le plus utilisé, nous le décrivons en détail dans la section suivante.

2.1.5 Perceptron multicouches

La structure du perceptron multicouches, ou MLP pour *Multi Layer Perceptron*, est définie par

- une **couche d'entrée**, qui n'est pas constituée de neurones, mais des composantes du vecteur des entrées,
- une ou plusieurs **couches** dites **cachées**, avec un nombre de neurones à déterminer selon le problème à résoudre.
- une **couche de sortie**, avec un neurone pour chaque composante de la variable de sortie du réseau.

On choisit la même fonction d'activation pour chaque couche du réseau : les neurones de la couche cachée utilisent une sigmoïde.

Réseau neuronal à deux couches Nous allons décrire la fonction définie par un perceptron à une seule couche cachée, qui est l'architecture retenue dans le code que nous utiliserons.

Le réseau neuronal présente un seul neurone de sortie, car dans le cadre de notre étude la fonction de modélisation est à valeurs dans $\mathbb{R}$, et un biais dans la couche d'entrée (cela implique une entrée supplémentaire constante égale à 1).

Le graphe de la figure V.7 décrit son fonctionnement.

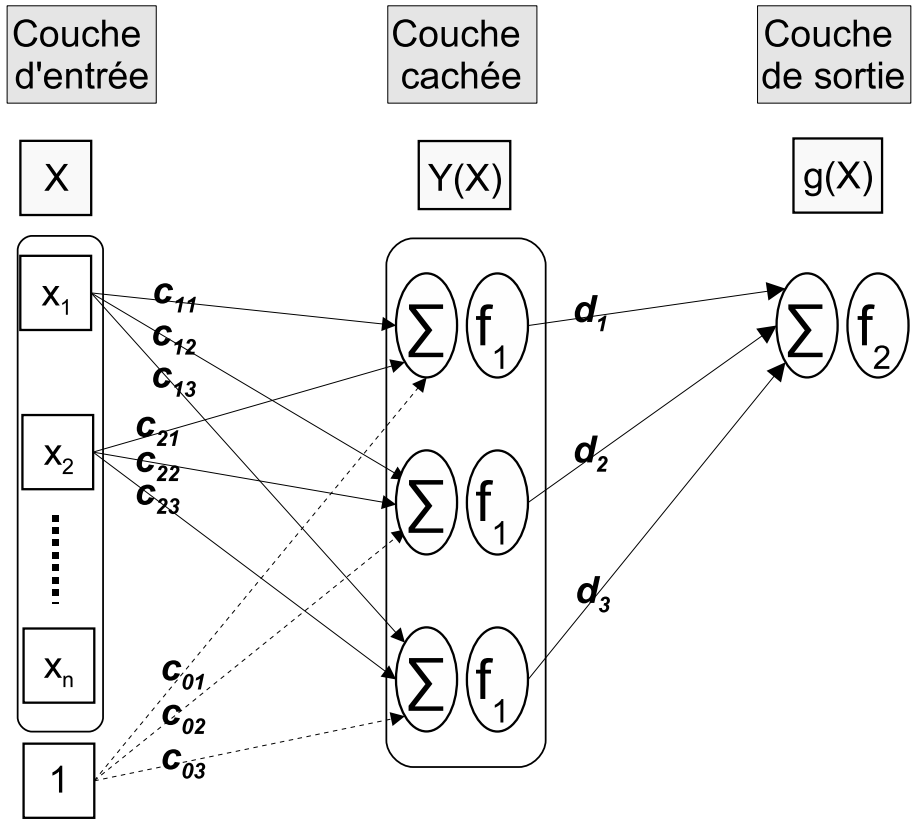


FIG. V.7 – Architecture et fonctionnement d'un réseau neuronal à deux couches

Ecrivons d'abord la réponse $g(X)$ du réseau neuronal en fonction de la sortie de la couche précédente

$$\begin{aligned} g(X) &= f_2\left(\sum_{j=1}^{m} d_j\ Y_j(X)\right) \\ &= f_2(d^T\ Y(X)) \end{aligned} \tag{V.47}$$

avec

- m le nombre de neurones de la couche cachée,
- $d \in \mathbb{R}^m$ le vecteur des poids entre la couche cachée et la couche de sortie,
- $Y(X) \in \mathbb{R}^m$ le vecteur obtenu après application de la couche cachée,
- f_2 la fonction d'activation appliquée à la sortie.

On obtient la relation entre les entrées et la sortie :

$$\begin{aligned} g(X) &= f_2\left[\sum_{j=1}^{m} d_j\ f_1(c_{0,j} + \sum_{i=1}^{n} c_{i,j}\ X_i)\right] \\ &= f_2\left[d^T\ f_1(c_0 + C^T\ X)\right] \end{aligned} \tag{V.48}$$

avec

- $X \in \mathbb{R}^n$ le vecteur des entrées,

- $C \in \mathbb{R}^{n \times m}$ la matrice des poids entre les entrées x_i et la couche cachée, et $c_0 \in \mathbb{R}^m$ le vecteur des poids pour le biais,
- f_1 la fonction d'activation appliquée à la couche cachée, avec un abus de notation dans l'écriture matricielle, car f_1 s'applique alors à toutes les composantes du vecteur $Y(X) = c_0 + C^T\ X$.

2.1.6 Propriété d'approximation universelle

On a montré dans [Hor89] que les réseaux de neurones sont des **approximateurs universels**, au sens où toute fonction à valeurs réelles bornée suffisamment régulière peut être approchée, avec une précision fixée, par un réseau de neurones non bouclé, pourvu que celui-ci possède une architecture appropriée :

- une couche cachée avec un nombre suffisant de neurones possédant la même fonction d'activation,
- un neurone de sortie linéaire.

Les perceptrons et les réseaux de neurones de type RBF ou à ondelettes possèdent cette propriété, mais en pratique, les perceptrons sont beaucoup plus utilisés car plus simples à mettre en oeuvre.

De plus la non-linéarité locale des RBF et ondelettes ne possède pas de bonnes propriétés en vue d'une utilisation pour l'optimisation.

2.2 Apprentissage

Pour le type de réseau neuronal que nous avons choisi, le processus appellé apprentissage cherche à obtenir la meilleure approximation du modèle G à partir de l'information fournie par les données et consiste

- à **ajuster les poids**,
- et **déterminer le nombre de neurones de la couche cachée**.

2.2.1 Sur-apprentissage

On montre (voir [Bar93]) que l'écart entre l'approximation du réseau neuronal et la fonction de modélisation est inversement proportionnel au nombre de neurones cachés.

Cette propriété n'est pas constructive, et il n'existe pas de résultat théorique pour prévoir le nombre de neurones cachés : il faut mettre en oeuvre une procédure numérique.

Ainsi, il ne faut pas fixer une précision sur les résidus des données d'apprentissage trop petite, sinon le nombre de neurones cachés augmente considérabement et on rencontre le problème du sur-apprentissage (voir section 1.2.5 page 186 pour le problème plus général du sur-ajustement).

En effet, pour un même ensemble de données, si on augmente le nombre de neurones de la couche cachée, l'erreur sur les données d'apprentissage se réduit car le réseau neuronal est plus riche, par contre l'erreur de généralisation augmente à cause de variations dépourvues de signification entre les données (phénomène d'oscillations).

2.2.2 Couples d'apprentissage

Les **données** de construction, ou d'**apprentissage**, sont définies par :

- un ensemble de points $\{X_1, \ldots, X_N\}$ issu d'un plan d'expériences,
- les évaluations correspondantes $\{G(X_1), \ldots, G(X_N)\}$, appelées **cibles**.

Les couples $(X_i, G(X_i))$ sont appelés **couples d'apprentissage**.

2.2.3 Apprentissage supervisé

Principe Le schéma de la figure V.8 présente le principe de l'**apprentissage supervisé** pour un nombre de neurones cachés fixé.

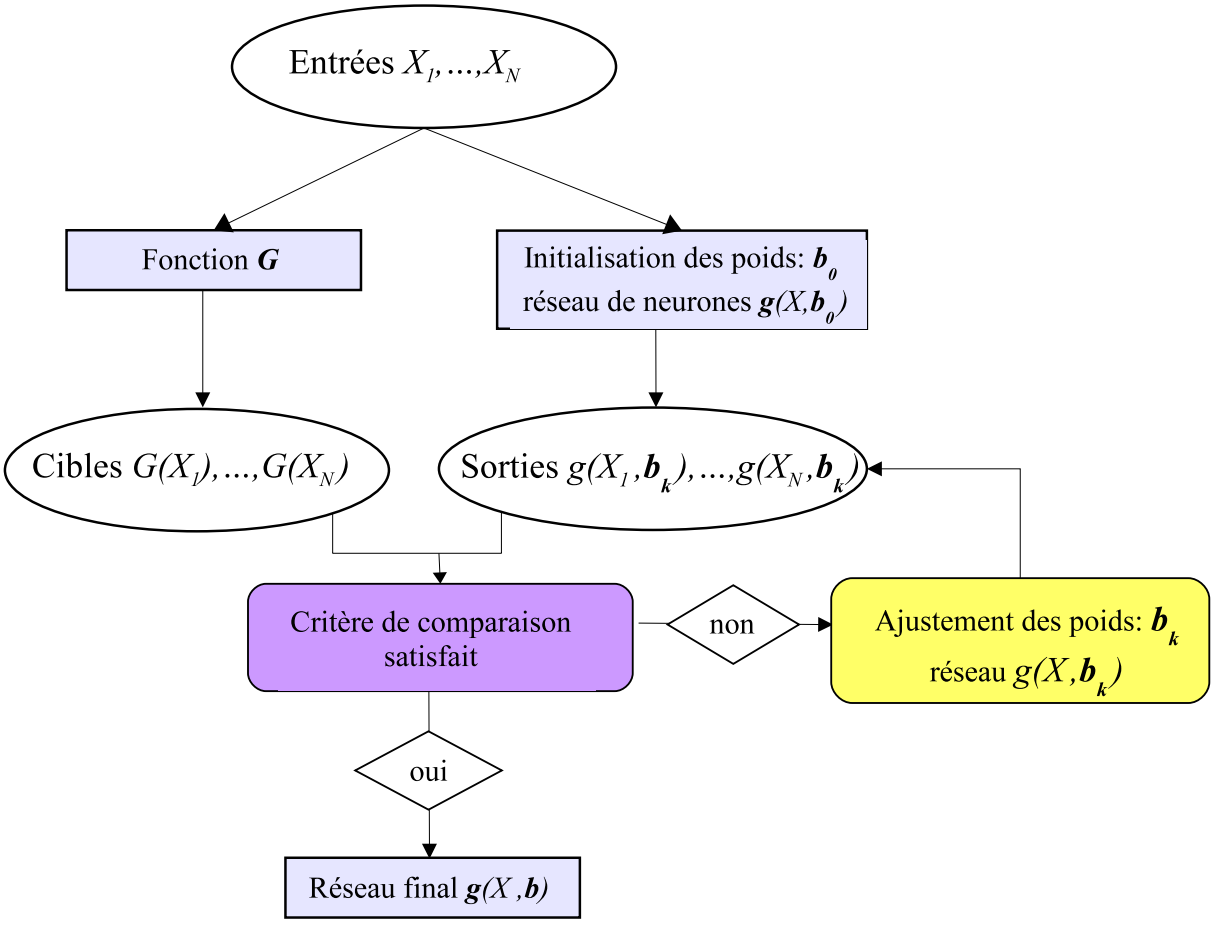


FIG. V.8 – Principe de l'apprentissage supervisé

Le principe consiste à suivre le processus itératif suivant :

- on se donne une tolérance $r > 0$,
- les poids $b \in \mathbb{R}^p$ sont initialisés avec $b^k = b_0$,
- on calcule l'écart e_k entre le réseau neuronal $g(., b^k)$ et le modèle sur les données d'apprentissage :
  - si $e_k > r$, on ajuste les poids b^k et on revient à l'étape précédente,
  - si $e_k < r$, on pose $b = b^k$ et $g(., b)$ est le réseau neuronal finalement obtenu.

Remarque Il existe aussi un apprentissage non supervisé, utilisé en analyse descriptive de données, quand on ne dispose pas de modèle pour calculer les couples d'apprentissage.

On cherche à regrouper de grands ensembles de données selon des critères de ressemblance ou proximité inconnus à priori (méthode "d'agrégation" ou "*clustering*" en statistiques). Les réseaux de neurones de type "cartes de Kohonen" s'organisent lors de la phase d'apprentissage pour donner une représentation dans un espace de petite dimension (typiquement deux) qui conserve les proximités entre données.

Le réseau neuronal à deux couches : une combinaison linéaire de fonctions de base Pour un nombre m de neurones cachés fixé, nous définissons le vecteur des poids $b := (c, d) \in \mathbb{R}^p$ avec

- $c \in \mathbb{R}^{(n+1).\ m}$ le vecteur regroupant les poids entre les entrées x_i et la couche cachée,
- $d \in \mathbb{R}^m$ le vecteur regroupant les poids entre la couche cachée et la couche de sortie,
- $p := (n+2).\ m$ le nombre total de poids à ajuster.

D'après la formulation $g(x) = f_2\left[d^T\ Y(X)\right]$ avec $Y(X)$ le vecteur de sortie de la couche cachée (voir section 2.1.5 page 208), comme f_2 est une fonction d'activation linéaire, en fait on peut considérer que $g(X)$ s'écrit comme une combinaison linéaire des $Y_i(X)$:

$$g(x) = \sum_{i=1}^{m} d_i\ \widetilde{Y}_i(X) \tag{V.49}$$

avec $\widetilde{Y}_i(X) = k_i\ Y_i(X)$ un multiple de la sortie du neurone caché n°i.

On peut donc considérer que le vecteur c permet la définition d'autant de fonctions $Y_i(X)$ que de neurones cachés. On peut interpréter cette relation dans le sens que les Y_i, non linéaires en X et en c, jouent le rôle de fonctions de base, et la sortie du réseau neuronal est une combinaison linéaire de ces fonctions de base avec pour coefficients les poids d_i.

Problème d'optimisation Nous allons détailler l'apprentissage supervisé dans le cadre du réseau neuronal à deux couches présenté en section 2.1.5 (voir page 208) en reprenant les mêmes notations.

Nous avons vu qu'en plus des poids il nécessaire d'ajuster aussi le nombre de neurones cachés.

Dans la phase d'apprentissage, les poids sont des variables, ainsi le réseau neuronal est une fonction qui s'écrit

$$\begin{array}{rccc} g: & \mathbb{R}^n \times \mathbb{R}^p & \longrightarrow & \mathbb{R} \\ & (X, b) & \mapsto & g(X, b) \end{array} \tag{V.50}$$

avec $b = (c\ d)^T$ l'ensemble des poids du réseau neuronal à déterminer.

Le critère qui mesure l'écart entre le réseau de neurones g et le modèle G sur les données d'apprentissage $(X_1, \ldots, X_N)$ est celui des moindres carrés :

$$\sum_{i=1}^{N} \|g(X_i, b) - G(X_i)\|^2. \tag{V.51}$$

On veut faire diminuer le critère en ajustant b, cela revient à résoudre le problème d'optimisation avec la fonction coût à minimiser :

$$\min_{b \in \mathbb{R}^p} J(b) := \frac{1}{2} \sum_{i=1}^{N} \|g(X_i, b) - G(X_i)\|^2. \tag{V.52}$$

C'est un problème d'optimisation d'une fonction non linéaire, on va chercher à le résoudre avec les méthodes performantes de type gradient : il vaut mieux choisir des fonctions d'activation dérivables.

Le gradient de la fonction coût se calcule à partir de celui du réseau neuronal g :

- directement par application des formules de dérivation de fonctions composées,
- de manière plus économique avec l'algorithme "de rétropropagation", qui consiste simplement à organiser le calcul différemment, en parcourant le graphe des connexions de la sortie vers l'entrée.

2.3 Mise en oeuvre

2.3.1 Logiciel GAP

Nous allons détailler le principe de fonctionnement du code de réseau de neurones que nous avons utilisé.

Il provient du travail de [VG04] sous la direction de M. Masmoudi et S. Jan au laboratoire MIP (pour Mathématiques pour l'Industrie et la Physique) de l'IMT (pour Institut de Mathématiques de Toulouse), qui a débouché sur un logiciel nommé GAP :

- La structure d'un **réseau neuronal à deux couches**, qui possède la propriété d'approximation universelle, est celle qui a été retenue.

- L'**apprentissage** utilise la méthode d'optimisation de **Levenberg-Marquardt** (méthode de Gauss-Newton avec conditionnement).
- Le code a été appliqué avec succès à des cas tests industriels pour des problèmes d'optimisation.

2.3.2 Fonctions sigmoïde

Les fonctions d'activation sont

- une fonction linéaire pour le neurone de sortie,
- une fonction sigmoïde modifiée afin que les poids ne deviennent pas trop grands, pour les neurones cachés.

2.3.3 Algorithme d'optimisation

L'algorithme d'optimisation de Levenberg-Marquardt a été présenté en section 3.3.3 de la première partie : c'est une méthode de descente dérivée de la méthode de Newton.

Nous rappelons le principe de cet algorithme.

Nous notons le problème d'optimisation

$$\min_{b\in\mathbb{R}^p} J(b) := \frac{1}{2}\|R(b)\|^2 \tag{V.53}$$

avec la fonction vectorielle des résidus

$$\begin{array}{rccc} R: & \mathbb{R}^p & \longrightarrow & \mathbb{R}^N \\ & b & \mapsto & R(b) := (g(X_1,b)-G(X_1),\ldots,g(X_N,b)-G(X_N))\,. \end{array} \tag{V.54}$$

L'algorithme de Newton est une méthode itérative : il consiste à résoudre le système suivant pour trouver la direction de descente d_k avec l'itéré courant b_k

$$\nabla^2 J(b_k)\ d_k = -\ \nabla J(b_k). \tag{V.55}$$

Or le gradient de J prend la forme

$$\nabla J(b) = DR(b)^T\ R(b) \tag{V.56}$$

avec $DR(b) \in \mathbb{R}^{N\times p}$ la matrice jacobienne de R en b.

Et la hessienne s'écrit

$$\nabla^2 J(b) = DR(b)^T\ DR(b) + \sum_{i=1}^{N} R_i(b)\ \nabla^2 R_i(b). \tag{V.57}$$

La méthode de Gauss-Newton consiste à considérer une approximation de la hessienne de la fonction coût J : le second terme de la dernière expression étant coûteux, on garde le premier terme plus simple à calculer (de plus l'approximation est justifiée à proximité de la solution car alors les R_i sont petits).

On obtient au rang k en remplaçant la hessienne dans le système (V.55)

$$DR(b_k)^T \; DR(b_k) \; d_k = - \; DR(b_k)^T \; R(b_k). \tag{V.58}$$

Etant donné que la matrice $DR(b_k)^T \; DR(b_k)$ est parfois mal conditionnée, l'algorithme de Levenberg-Marquardt apporte une correction en remplaçant la matrice précédente par

$$DR(b_k)^T \; DR(b_k) + \rho \; I_N \tag{V.59}$$

avec $\rho \in \mathbb{R}^{+*}$ qui permet de rendre le système linéaire plus stable sans changer la solution de (V.55).

Finalement, on est conduit à résoudre le système suivant pour trouver la direction de descente d_k

$$\left(DR(b_k)^T \; DR(b_k) + \rho \; I_N\right) \; d_k = - \; DR(b_k)^T \; R(b_k). \tag{V.60}$$

La matrice du membre de gauche est symétrique, et le système linéaire est résolu par la méthode du gradient conjugué (voir par exemple [BGLS03]).

Cette méthode comporte l'avantage de ne nécessiter que la connaissance du produit de la matrice par un vecteur : lors de l'implémentation, on évite de stocker la matrice $DR(b_k)^T \; DR(b_k)$ de grande taille, ce qui améliore l'efficacité de l'algorithme.

2.3.4 Différenciation algorithmique

Le calcul du produit $DR(b)^T \; DR(b) \; d_k$ s'effectue donc sans calculer les matrices jacobiennes, et en utilisant la différenciation algorithmique : cette technique a été introduite pour calculer de manière efficace des dérivées dans le cas d'une fonction définie par un code de calcul (voir [Gri00]).

Le principe est de dériver chaque ligne du code en utilisant les règles de dérivation des fonctions composées

- soit du début à la fin pour la dérivation dite en **mode direct**, adaptée pour dériver une fonction vectorielle par rapport à une seule variable,
- soit de la fin au début pour la dérivation dite en mode indirect ou **mode inverse**, adaptée pour dériver une fonction réelle par rapport à plusieurs variables.

Ici on calcule

- $z := DR(b) \; d = \partial_\varepsilon R(b + \varepsilon, d)|_{\varepsilon=0}$ en mode direct,
- puis $DR(b)^T \; z = D\left(z^T \; R(b)\right)$ en mode indirect.

2.3.5 Régularisation

Motivation

Eviter les minima locaux et les petites variations Les modèles à approcher sont parfois fortement non linéaires, l'idée de la régularisation, déjà évoquée en section 1.5.2 (voir page 198) est de "lisser" les comportements de la fonction.

En effet, pour une utilisation ultérieure dans un problème d'optimisation, la régularisation appliquée au réseau neuronal permet d'éviter les minima locaux.

Une illustration de cette propriété est présentée dans la figure V.9 : un réseau de neurones a été construit avec cette méthode pour approcher la fonction test Rastrigin en dimension deux.

Cette fonction est une fonction carré "bruitée" : sa surface a l'allure générale de la parabole de fonction carré mais possède de nombreux minima locaux (surface dessinée en transparence) qui piègent généralement les algorithmes qui cherchent le minimum global.

Le réseau neuronal obtenu (3 neurones cachés et 12 expériences ont été nécessaires) a éliminé ces variations indésirables (surface dessinée en couleur) et permet résoudre facilement le problème d'optimisation avec une méthode de descente.

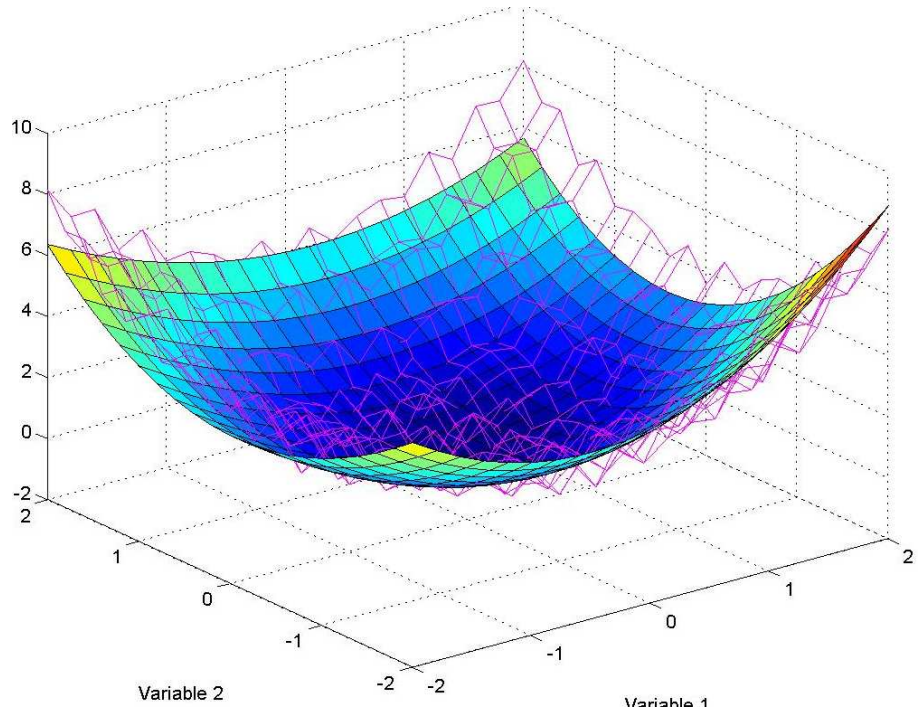


FIG. V.9 – Exemple de réseau de neurones approximation de la fonction Rastrigin

De plus, nous utiliserons le gradient du réseau neuronal pour estimer les paramètres importants associés aux plus fortes variations globales de la fonction de modélisation. Dans cet objectif, il est souhaitable que l'approximation g ne "capture pas" les variations locales, ce que permet la régularisation.

Qualité de l'apprentissage La difficulté de l'apprentissage va venir du fait qu'il ne faut pas trop enrichir le réseau neuronal avec un nombre trop important de neurones cachés pour éviter le sur-apprentissage (voir section 2.2.1 page 210), et que l'objectif est d'obtenir une bonne qualité de prédiction.

Il faut trouver le **compromis** entre un **bon apprentissage** mesuré par le critère d'erreur sur les données d'apprentissage et une **bonne généralisation** mesurée par le critère d'erreur sur les données de validation.

Techniques usuelles Dans le cadre du problème de minimisation des résidus pour l'apprentissage d'un réseau de neurones, nous présentons les techniques usuelles de régularisation.

Régularisation par Gauss-Newton L'algorithme d'optimisation utilise la méthode de Gauss-Newton : cette méthode calcule des **perturbations de norme minimale** (voir [Mas87]), ce qui entraîne une régularisation de fait.

Ainsi, en initialisant les coefficients c de la première couche, qui définissent les fonctions de base Y_i du réseau neuronal (voir section 2.2.3 page 212), avec une petite norme, nous choisissons des fonctions Y_i régulières. La méthode de Gauss-Newton permet de garder la norme de c petite, donc aussi la régularité des Y_i.

Régularisation par réduction du nombre de neurones cachés Si le nombre de neurones cachés est trop important, le sur-apprentissage peut se produire malgré l'application de techniques de régularisation.

Si le nombre de neurones cachés est trop faible, le réseau neuronal ne pourra pas approcher correctement la fonction de modélisation.

En pratique, la qualité de parcimonie (voir section 1.2.5 page 187) est donc essentielle, et choisir le nombre minimum de neurones cachés nécessaires peut être vu comme une technique de régularisation.

L'algorithme d'apprentissage utilisera donc un nombre initial de neurones cachés petit, puis on augmentera ce nombre jusqu'à obtenir un réseau neuronal assez riche en fonctions de base pour produire une approximation correcte.

Régularisation par arrêt des itérations Poursuivre trop loin le processus de minimisation des résidus sur le lot d'apprentissage peut dégrader la qualité de prédiction à cause du sur-apprentissage : la technique de régularisation consiste alors à arrêter le processus d'apprentissage avant que n'apparaissent les oscillations (*stopped training method*, voir [CLG00]).

En pratique, au fur et à mesure des itérations, l'erreur sur le lot d'apprentissage diminue, puis se stabilise, alors que l'erreur sur le lot de vérification diminue, puis augmente quand apparaît le sur-apprentissage : il faut arrêter le processus quand cette dernière erreur commence à augmenter.

Régularisation de Tikhonov Cette technique, qui consiste à ajouter un terme à la fonction coût des résidus à minimiser, a été présentée en section 1.5.2 (voir page 198).

Dans le cas du code de réseau de neurones utilisé, le terme est la norme des coefficients définissant les fonctions de base, notés c, (voir section 2.2.3 page 212).

L'idée est la même que pour la régularisation par Gauss-Newton précédente : garder la norme de c petite afin d'éviter de construire des fonctions de base trop raides, qui entraînent des oscillations de l'approximation.

La fonction coût régularisée s'écrit

$$J_r(b) = J(b) + \sigma \ \|c\|^2 \tag{V.61}$$

avec

- c les poids entre les entrées et la couche cachée,
- $J(b) := \frac{1}{2} \sum_{i=1}^{N} (g(X_i, b) - G(X_i))^2$ la norme au carré des résidus sur le lot d'apprentissage (voir section 2.3.3 page 214),
- $\sigma \in \mathbb{R}^+$ une constante.

La régularisation n'est pas effectuée sur les poids notés d entre la couche cachée et la couche de sortie.

On obtient alors le système linéaire associé à la fonction coût régularisée pour trouver la direction de descente $d = (d_1 \ d_2)^T$ au rang k :

$$\left[DR(b_k)^T \ DR(b_k) + \rho \ I_N + \sigma \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} \right] \begin{pmatrix} d_1 \\ d_2 \end{pmatrix} = - \ DR(b_k)^T \ R(b_k) - \sigma \begin{pmatrix} c_k \\ 0 \end{pmatrix}. \tag{V.62}$$

2.3.6 Algorithme

Les techniques de régularisation précédentes ont été implémentées dans le logiciel GAP (voir [VG04]).

L'apprentissage suit alors l'algorithme présenté page 219.

Algorithme de construction du réseau de neurones

▷ **Entrées**

- le scénario étudié, avec les paramètres incertains et les lois de probabilité associées,
- N le nombre de données de construction du réseau neuronal, qui mesure le coût de calcul associé à la construction de l'approximation.
- la tolérance τ définissant la précision.

▷ **Construction des données**

- Construction des trois lots de données $L_i = \{X_1, \ldots, X_{N_i}\}$, $i = 1, 2, 3$, obtenus par trois tirages de type **hypercube latin** indépendants, avec
  - pour L_1 le lot d'apprentissage et L_2 le premier lot de vérification, $N_1 + N_2 := N$.
    Nous fixons une proportion par rapport à N, par exemple $N_1 = 80\%\ N$ et $N_2 = 20\%\ N$.
  - L_3 est le deuxième lot de vérification, que nous fixons unique pour tous les réseaux neuronaux construits, de taille $N_3 = 1000$ par exemple.
    Cela permet de calculer un seul tirage, et de comparer les erreurs de vérification des différents réseaux neuronaux sur les mêmes données.
- Calcul des cibles $G(X_i)$ associées aux données précédentes.

▷ **Premier apprentissage : calcul du nombre m de neurones cachés**

- $\sigma = 0$ (pas de régularisation)
- initialisation du nombre de neurones cachés : $k = 1$
- initialisation des poids : $b^k := b_0$
- Tant que $\|R(b^k)\| > \tau$
  - minimiser sur le lot L_1 : $J(b) := \frac{1}{2} \sum_{i=1}^{N_1} \left(g(X_i, b) - G(X_i)\right)^2$, on obtient b^k
  - $k := k + 1$

▷ **Régularisation** Avec m le nombre de neurones cachés déterminé à la phase précédente :

- minimiser sur le lot L_1 : $J_r(b) = J(b) + \sigma\ \|b\|^2$ avec plusieurs valeurs de σ,
- choisir la valeur de σ qui minimise les résidus sur le lot L_2 : $\frac{1}{2} \sum_{i=1}^{N_2} \left(g(X_i, b) - G(X_i)\right)^2$

▷ **Deuxième apprentissage**

- initialiser b avec les poids obtenus à la fin de l'apprentissage de la phase précédente qui a permis de fixer σ le coefficient de régularisation,
- minimiser sur les lots $L_1 \cup L_2$: $J(b) := \frac{1}{2} \sum_{i=1}^{N_1+N_2} \left(g(X_i, b) - G(X_i)\right)^2$.

▷ **Valisation**

Calculer les erreurs sur le lot de vérification L_3 pour valider le réseau neuronal obtenu.

TAB. V.1 – Algorithme de construction du réseau de neurones

3 *Sparse grids*

3.1 Principe

3.1.1 Introduction

La méthode d'interpolation dite ***sparse grid*** utilise des fonctions bases (linéaires par morceaux ou polynomiales) hiérarchiques, définies sur les points d'une grille éparse (nombre de points réduit par rapport aux méthodes d'interpolation classiques).

Les *sparse grids* entrent ainsi dans la catégorie des méthodes d'approximation parcimonieuses, adaptées aux problèmes en grande dimension.

Le principe des *sparse grids* est d'étendre au cas multivarié les formules d'interpolation connues en dimension un.

L'interpolation *sparse grid* utilise la construction de Smolyak (voir [Smo63]). Elle permet de sélectionner une partie seulement des produits tensoriels qui définissent les fonctions de base multivariées : on définit une approche hiérarchique, dans laquelle les fonctions de base qui ont un support petit, donc qui contribuent peu à la représentation de la fonction à approcher, sont négligées.

Selon les caractéristiques de la fonction, des choix différents des fonctions de base et des points de grille associés sont possibles.

Nous utiliserons un code d'interpolation *sparse grid* (voir [Kli05]), qui présente l'avantage de proposer une option "adaptative" qui permet de travailler en dimension supérieure par rapport à l'algorithme classique.

L'idée est la suivante :

- la grille de départ est grossière,
- l'algorithme détecte les directions importantes,
- il raffine automatiquement uniquement dans ces directions.

Applications Cette méthode d'interpolation a été appliquée avec succès à la résolution d'équations différentielles (voir [Zen91], [Gri91], [Ach03]), au traitement d'images, à la compression de données, à la classification (*data mining*), à l'intégration numérique (voir [FHP96]) ou aux problèmes d'approximation (voir [GK00]).

Approximation unique Ici les données sont déterminées de manière analytique (les méthodes d'interpolations n'utilisent pas de plans d'expérience) : pour un modèle G fixé, l'approximation g obtenue est déterminée de manière unique.

Dans le cas des réseaux de neurones, pour un modèle et des données d'apprentissage fixées, selon le résultat de l'optimisation de la phase d'apprentissage, l'approximation obtenue est différente.

Le détail des propriétés sur les *sparse grids* est exposé dans [Kli05], [BG04], [BNR00].

3.1.2 Code utilisé

Le code utilisé pour générer les *sparse grids* dans notre étude provient de Klimke (voir [Kli05]), sous la forme d'une *toolbox* pour Matlab.

3.2 Méthode d'interpolation

3.2.1 Dimension un

Formule d'interpolation Nous considérons une fonction de modélisation

$$G : \quad [0,1] \quad \longrightarrow \quad \mathbb{R} \tag{V.63}$$

et son approximation par interpolation g.

La formule d'interpolation en dimension un s'écrit

$$g_i(G) := \sum_{x_j^i \in X^i} a_j^i \,.\, G(x_j^i) \tag{V.64}$$

avec

- $g_i(G)$ l'interpolation de niveau i de la fonction G, notée g_i quand il n'y a pas de confusion possible,
- $X^i = \{x_j^i \in [0,1],\ j = 1, \ldots, m_i\}$ l'ensemble des **points d'interpolation** ou **noeuds**,
- $\{a_j^i \in \mathcal{C}([0,1]),\ j = 1, \ldots, m_i\}$ les **fonctions de base** associées aux noeuds, telles que
  - $\forall j \in \{1, \ldots, m_i\} \quad a_j^i(x_j^i) = 1$,
  - $\forall (j,k) \in \{1, \ldots, m_i\}^2 \quad j \neq k \Rightarrow a_j^i(x_k^i) = 0$.

Chaque fonction de base a_j^i vaut un au noeud x_j^i associé, et zéro aux autres noeuds. Cette propriété assure le principe de l'interpolation : **la fonction et son interpolée sont égales sur les noeuds**

$$\forall x_j^i \in X^i \; g_i(x_j^i) = G(x_j^i). \tag{V.65}$$

Approche hiérarchique On calcule la différence notée Δ^i entre l'interpolation de niveau i et celle de niveau $i-1$ (avec les notations de la formule (V.64)).

On fait intervenir la propriété $g_{i-1}(G) = g_i(g_{i-1}(G))$, vraie pour les fonctions de base utilisées dans la suite, et on obtient

$$\begin{aligned}
\Delta^i &:= g_i(G) - g_{i-1}(G) \\
&= \sum_{x_j^i \in X^i} a_j^i \,.\, G(x_j^i) - \sum_{x_j^i \in X^i} a_j^i \,.\, g_{i-1}(x_j^i) \\
&= \sum_{x_j^i \in X^i} a_j^i \,.\, (G(x_j^i) - g_{i-1}(x_j^i)).
\end{aligned}$$

Nous supposons à présent que nous utilisons des fonctions de base vérifiant la propriété suivante, essentielle pour permettre de définir les fonctions de base hiérarchiques : **les noeuds d'un niveau sont inclus dans ceux du niveau supérieur**, ce qui s'écrit

$$X^{i-1} \subset X^i. \tag{V.66}$$

Comme de plus d'après le principe d'interpolation, pour tout $x_j^i \in X^{i-1} \quad G(x_j^i) - g_{i-1}(x_j^i) = 0$, alors on simplifie l'expression Δ^i :

$$\Delta^i = \sum_{x_j^i \in X_\Delta^i} a_j^i \,.\, (G(x_j^i) - g_{i-1}(x_j^i)) \tag{V.67}$$

avec

- $X_\Delta^i := X^i \backslash X^{i-1}$ les nouveaux noeuds rajoutés par le niveau i au niveau précédent,
- $m_i^\Delta = m_i - m_{i-1}$ est le nombre d'éléments de X_Δ^i.

Après avoir numéroté les x_j^i et les a_j^i pour avoir le j^{eme} élément de X_Δ^i noté x_j^i, on peut finalement écrire

$$\Delta^i = \sum_{x_j^i \in X_\Delta^i} a_j^i \,.\, w_j^i \tag{V.68}$$

avec $w_j^i := G(x_j^i) - g_{i-1}(x_j^i)$ le **surplus hiérarchique**, calculé au noeud x_j^i comme la différence entre G et l'interpolation g_{i-1} de niveau $i-1$.

Ainsi, pour obtenir Δ^i, on **utilise seulement les évaluations sur les noeuds n'appartenant pas aux niveaux précédents** afin de calculer les surplus hiérarchiques (voir illustration dans les figures V.10, V.11 et V.12).

Finalement, en utilisant le principe de somme télescopique, avec $g_0 = 0$ et $\Delta^i := g_i - g_{i-1}$, on peut exprimer g_i en fonction des Δ^i

$$g_i = \sum_{k=1}^{i} \Delta^k. \tag{V.69}$$

Fonctions de base Pour la construction d'une *sparse grid*, nous utiliserons des fonctions de base

- de type linéaire par morceaux ("Clenshaw-Curtis"),
- ou de type polynôme ("Chebyshev)".

Type Clenshaw-Curtis Les fonctions de base Clenshaw-Curtis sont définies en dimension un par

$$\begin{aligned} a_1^1(x) &:= 1 \\ a_j^i(x) &:= \begin{cases} 1 - (m_i - 1)\,|x - x_j^i| & \text{si } |x - x_j^i| < \frac{1}{m_i - 1} \\ 0 & \text{sinon} \end{cases} \quad \text{pour } i > 1 \text{ et } j = 1, \ldots, m_i. \end{aligned} \tag{V.70}$$

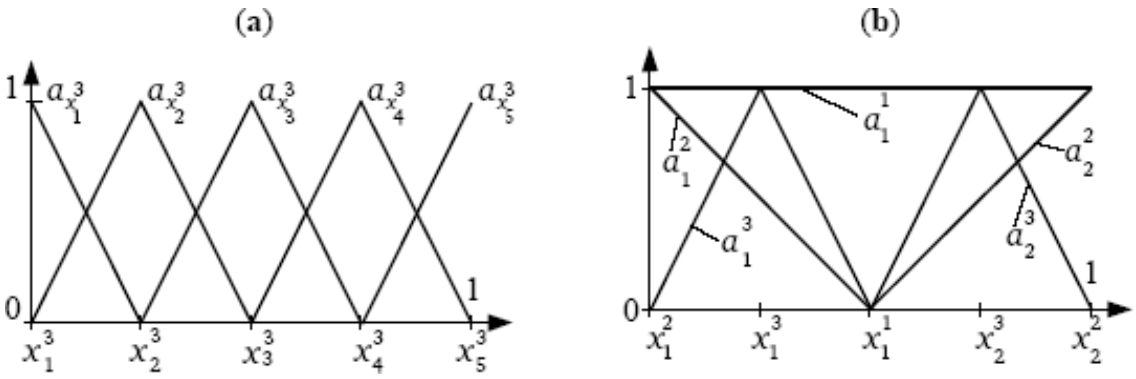


FIG. V.10 – Fonctions de base type Clenshaw-Curtis (dimension 1 - niveau 3)
A gauche : fonctions de base classiques pour grille pleine
A droite : fonctions de base hiérarchiques pour *sparse grid*

La figure V.10 présente les fonctions de base de niveau 3 associées à une grille pleine et des fonctions de base hiérarchiques pour la *sparse grid*.

Les noeuds de type Clenshaw-Curtis sont définis en dimension un par

$$x_j^i := \begin{cases} 0.5 & \text{pour } j = 1 \text{ et si } m_i = 1 \\ \frac{j-1}{m_i-1} & \text{pour } j = 1, \ldots, m_i \text{ si } m_i > 1 \end{cases} \tag{V.71}$$

avec

$$m_i := \begin{cases} 1 & \text{si } i = 1 \\ 2^{i-1}+1 & \text{si } i > 1. \end{cases} \tag{V.72}$$

Type Chebyshev Les fonctions de base Chebyshev sont des polynômes de Chebyshev, définis en dimension un par

$$\begin{aligned} a_1^1(x) &:= 1 \\ a_j^i(x) &:= \prod_{\substack{k=1 \\ k \neq j}}^{m_i} \frac{x - x_k^i}{x_j^i - x_k^i} \text{ pour } i > 1 \text{ et } j = 1, \ldots, m_i. \end{aligned} \tag{V.73}$$

La figure V.11 présente les fonctions de base de niveau 3 associées à une grille pleine et des foncions de base hiérarchiques pour la *sparse grid*.

Les fonctions de base de type Chebyshev sont de degré supérieur à celles de type Clenshaw-Curtis, on peut les utiliser pour obtenir une plus grande précision.

Les noeuds de type Chebyshev sont définis en dimension un par

$$x_j^i := \begin{cases} 0.5 & \text{pour } j = 1 \text{ et } m_i = 1 \\ \frac{1}{2}\left(-cos\left(\pi \frac{j-1}{m_i-1}\right) + 1\right) & \text{pour } j = 1, \ldots, m_i \end{cases} \tag{V.74}$$

avec comme précédemment

$$m_i := \begin{cases} 1 & \text{si } i = 1 \\ 2^{i-1}+1 & \text{si } i > 1. \end{cases} \tag{V.75}$$

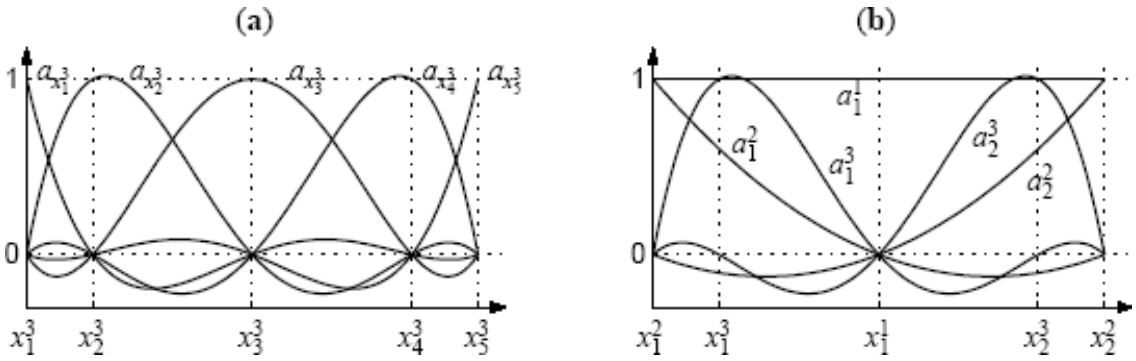


FIG. V.11 – Fonctions de base type Chebyshev (dimension 1 - niveau 3)
A gauche : fonctions de base classiques pour grille pleine
A droite : fonctions de base hiérarchiques pour *sparse grid*

Niveau	1	2	3	4
Nombre de noeuds	1	3	5	9

TAB. V.2 – Fonctions de base type Clenshaw-Curtis ou Chebyshev : nombre de noeuds dans une direction en fonction du niveau

Illustration Le tableau V.2 donne le nombre de noeuds utilisés par les fonctions de base type Clenshaw-Curtis ou type Chebyshev dans une direction en fonction du niveau.

La figure V.12 présente une illustration du principe d'interpolation hiérarchique des *sparse grids* avec les fonctions de base type Clenshaw-Curtis de niveau 3 :

- Dans le cas de l'interpolation classique, on utilise les cinq fonctions de base classiques :

  en chaque noeud on calcule directement les coefficients $G(x_i^3)$ associé aux fonctions de base correspondant aux noeuds x_i^j.

- Dans le cas de l'interpolation *sparse grid*, on utilise les cinq fonctions de base hiérarchiques :

  - on commence par calculer le coefficient notés w_1^1 du premier niveau (ici pour la fonction de base constante associée à x_1^1),
  - puis on passe aux coefficients du deuxième niveau (ici pour les deux fonctions de base associées à x_1^2 et x_2^2) :

    on ne calcule que les surplus, c'est-à-dire la différence entre l'interpolation de niveau un et G, notés w_1^2 et w_2^2,
  - on continue le processus avec les coefficients des fonctions de base des niveaux supérieurs.

Les **surplus hiérarchiques tendent vers zéro** quand le niveau de la *sparse grid* tend vers l'infini, ils sont **utilisés pour l'estimation et le contrôle de l'erreur** d'interpolation.

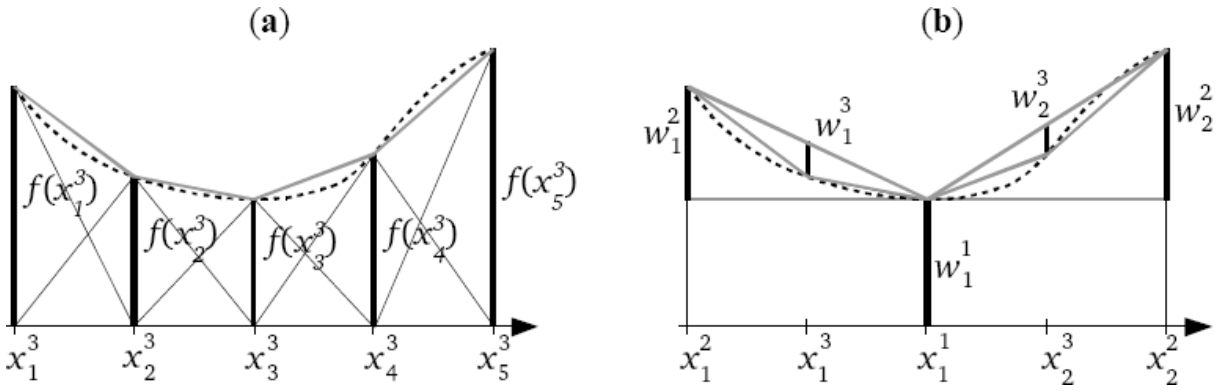


FIG. V.12 – Principe de l'interpolation (type Clenshaw-Curtis - dimension 1 - niveau 3)
A gauche : fonctions de base classiques pour grille pleine
A droite : fonctions de base hiérarchiques pour *sparse grid*

3.2.2 Dimension d

Nous considérons une fonction de modélisation

$$G: \quad [0,1]^n \quad \longrightarrow \quad \mathbb{R} \tag{V.76}$$

et son approximation par interpolation g.

Produit tensoriel Dans le cas de d fonctions à une variable réelles $f_1, \dots f_d$, le produit tensoriel est l'opération suivante qui permet de construire une fonction f de d variables réelle

$$f = f_1 \otimes \dots \otimes f_d \tag{V.77}$$

avec $f(X) = f(x_1, \dots x_d) := \prod_{j=1}^{d} f_j(x_j)$.

Fonctions de bases en dimension supérieure La figure V.13 montre un exemple de fonction de base linéaire par morceaux en dimension deux, obtenue comme produit tensoriel de deux fonctions de base en dimension un.

En dimension supérieure, le principe est le même : on obtient également les fonctions de base comme produit tensoriel des fonctions de base en dimension un.

Chaque point, ou noeud, de la grille d'interpolation est associé à une fonction de base différente dont il est le sommet.

Formule d'interpolation La formule d'interpolation en dimension d s'obtient en appliquant un produit tensoriel à la formule (V.64) (voir page 221), en supposant que nous pouvons avoir des résolutions différentes suivant chaque direction, c'est-à-dire des X^i différents

$$g := g_{i_1} \otimes \dots \otimes g_{i_d} = \sum_{x_{j_1}^{i_1} \in X^{i_1}} \dots \sum_{x_{j_d}^{i_d} \in X^{i_d}} (a_{j_1}^{i_1} \otimes \dots \otimes a_{j_d}^{i_d}) \cdot G(x_{j_1}^{i_1}, \dots, x_{j_d}^{i_d}). \tag{V.78}$$

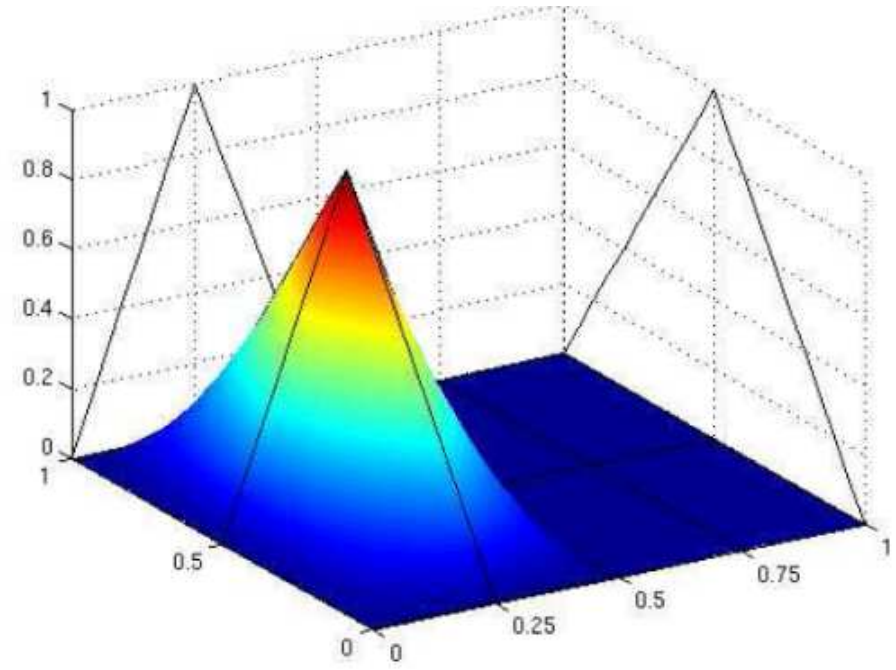


FIG. V.13 – Exemple de fonction de base en dimension 2 (type Clenshaw-Curtis)

L'interpolation classique est donc définie pour des noeuds remplissant une grille pleine de $m_{i_1} \times \ldots \times m_{i_d}$ points : ce nombre augmente de façon exponentielle avec la dimension d et devient rapidement impraticable en pratique.

Vecteurs d'indices Nous notons

- $\mathbf{i} = (i_1, \ldots, i_d) \in \mathbb{N}^d$ un vecteur d'indices représentant la résolution de la grille,
- $|\mathbf{i}|_1 := \sum_{k=1}^{d} i_k$ sa norme L_1,
- $|\mathbf{i}|_\infty := max\{i_1, \ldots, i_d\}$ sa norme L_∞.

Ecriture hiérarchique En utilisant l'approche hiérarchique (V.69) (voir page 222), cette formule s'écrit en fonction des Δ^i

$$g = \sum_{i_1=1}^{m_{i_1}} \ldots \sum_{i_d=1}^{m_{i_d}} (\Delta^{i_1} \otimes \ldots \otimes \Delta^{i_d}) \tag{V.79}$$

avec $\Delta_i := g_i - g_{i-1}$ et $g_0 = 0$.

Construction de Smolyak Pour définir les fonctions de base hiérarchiques utilisées par la *sparse grid*, on utilise la construction de Smolyak (voir [Smo63]) qui permet sélectionner une partie seulement de tous les produits tensoriels.

Formule d'interpolation *sparse grid* La fonction d'interpolation *sparse grid* g_n de rang n s'écrit cette fois

$$g_n := \sum_{|\mathbf{i}|_1 \leqslant d+n} (\Delta_{i_1} \otimes \ldots \otimes \Delta_{i_d}) \tag{V.80}$$

avec

- $n \in \mathbb{N}$ tel que $d \leqslant d+n \leqslant m_{i_1} + \ldots + m_{i_d}$,
- $\Delta_i := g_i - g_{i-1}$ et $g_0 = 0$.

En gardant dans la somme (V.79) uniquement les produits tensoriels correspondant aux indices vérifiant $|\mathbf{i}|_1 \leqslant d+n$, la construction de Smolyak élimine les produits tensoriels des fonctions de base avec les supports les plus petits (voir illustration suivante) : pour réduire le nombre de points de grille en détériorant le moins possible la précision, on néglige les contributions à la construction de l'approximation les plus faibles.

Illustration La figure V.14 représente une grille pleine de rang 3 (correspondant à $m_1 = m_2 = 9$ dans chaque direction) en dimension deux (81 points pour le type Clenshaw-Curtis).

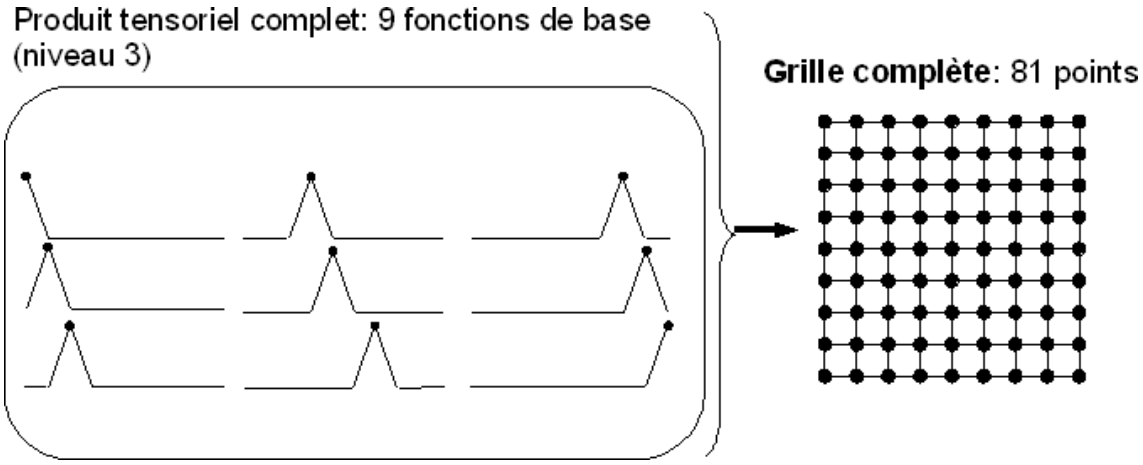


FIG. V.14 – Grille pleine (type Clenshaw-Curtis - dimension 2 - rang 3)

On visualise ensuite la *sparse grid* correspondante (rang 3 en dimension deux : 29 points) :

- Sur la figure V.15 sont représentées les fonctions de bases en dimension un, les produits tensoriels non sélectionnés en pointillés, et les sommets et supports des fonctions de base.
- Sur la figure V.16 sont représentées les mêmes fonctions de base hiérarchiques sous forme de graphes.

Dans le cas où un même pas de discrétisation est utilisé dans toutes les directions, une grille pleine correspond à l'utilisation de la norme L_∞ dans la somme (V.79)

$$g_q := \sum_{|\mathbf{i}|_\infty \leqslant n+1} (\Delta_{i_1} \otimes \ldots \otimes \Delta_{i_d}). \tag{V.81}$$

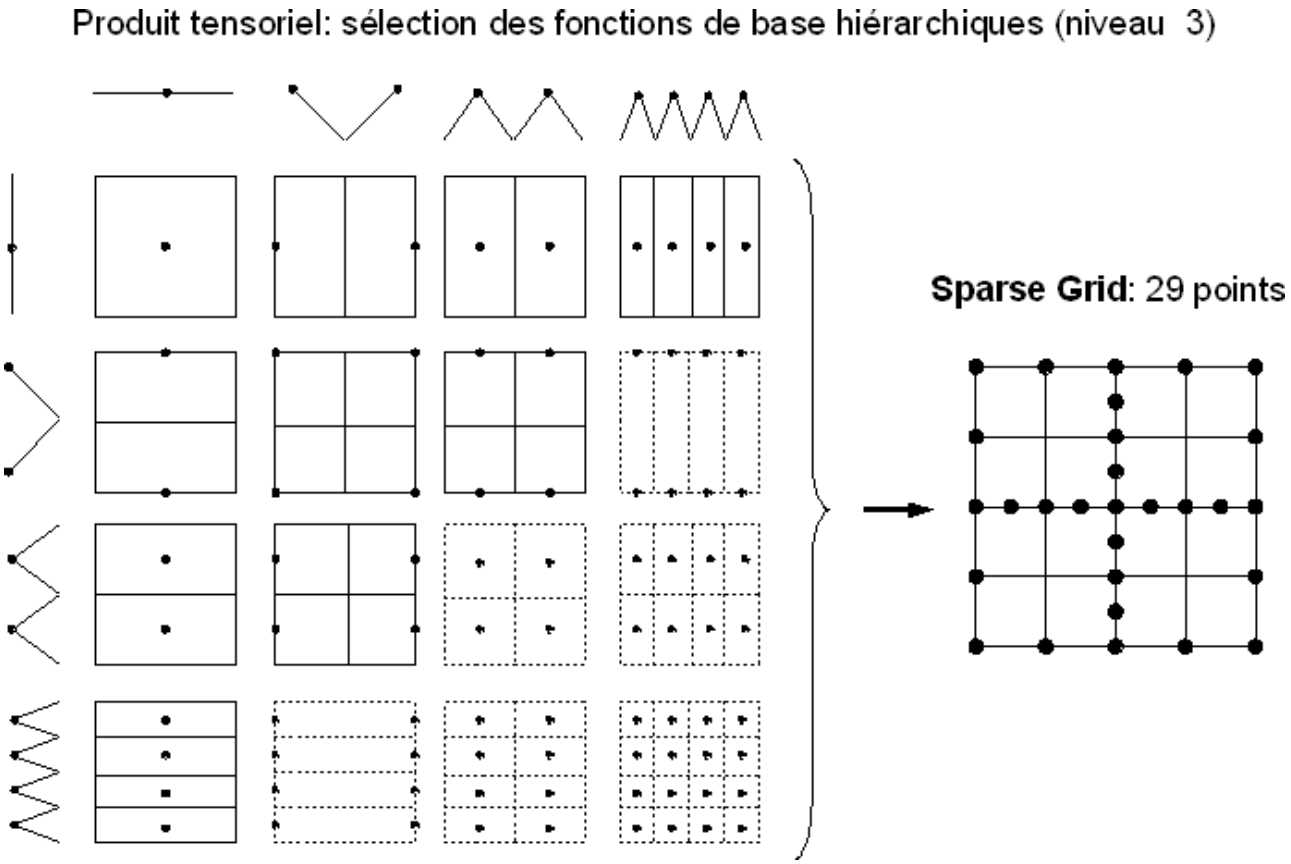


FIG. V.15 – *Sparse grid*, fonctions de base hiérarchiques : supports et sommets (type Clenshaw-Curtis - dimension 2 - rang 3)

Construire la *sparse grid* consiste à passer de la norme L_∞ à la norme L_1, ce qui se traduit par l'élimination dans la figure V.15 des fonctions de base représentées dans le triangle inférieur droit, correspondant aux fonctions de base à petit support.

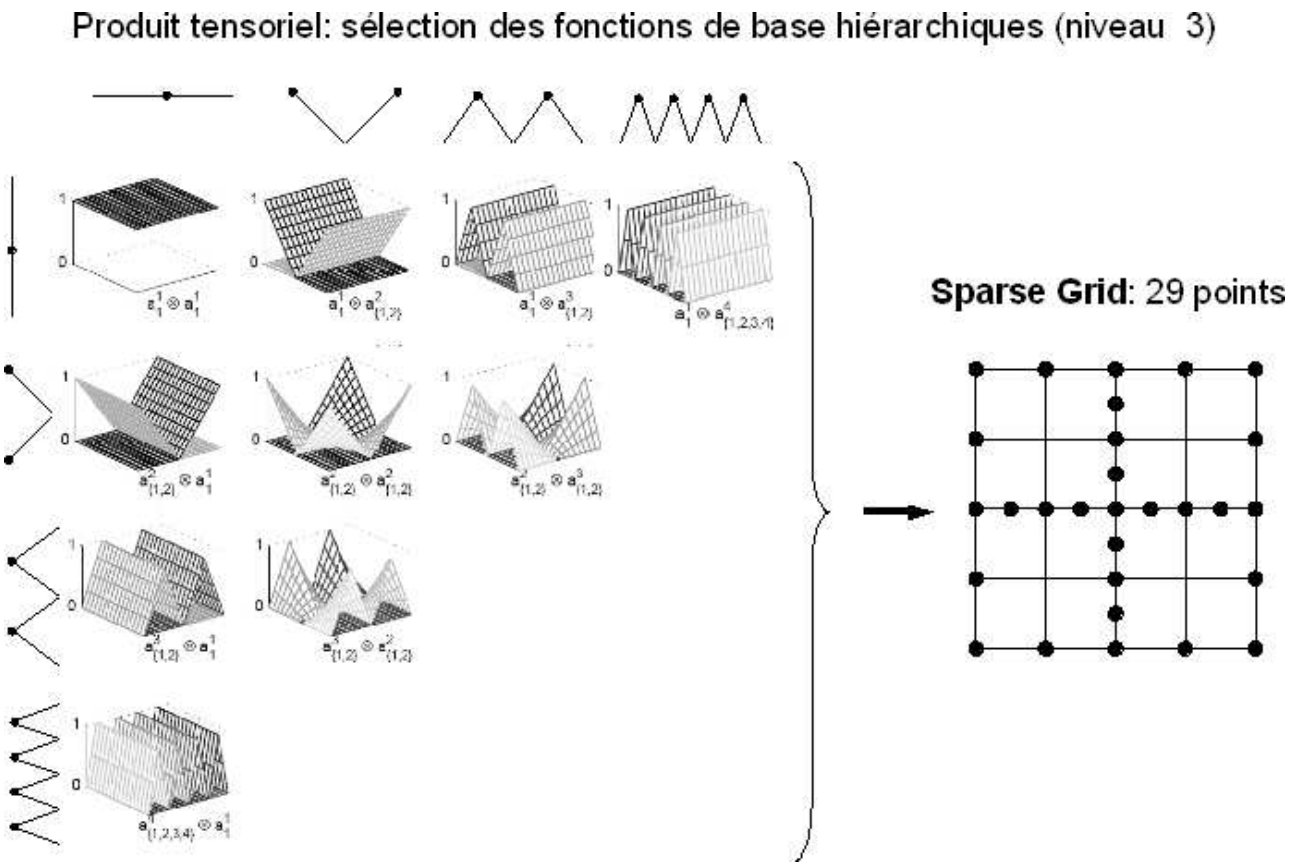


FIG. V.16 – *Sparse grid*, fonctions de base hiérarchiques : graphes (type Clenshaw-Curtis - dimension 2 - rang 3)

3.2.3 Autres types de *sparse grid*

Il existe d'autres types de *sparse grids*, construites avec des fonctions de base différentes : le nombre de points de grille et leur répartition varie (voir [BG04], [Spr98]).

La figure V.17 montre les quatre types de *sparse grids* proposées par le code utilisé :

- celles de type Clenshaw-Curtis, "*Maximum Norm*" et "*No Boundary*" utilisent des fonctions de base linéaires par morceaux,
- celle de type Chebyshev est la seule à utiliser des fonctions de base non linéaires (polynômes).

Selon les caractéristiques de la fonction à approcher, un certain type de *sparse grid* peut être plus performant.

Le nombre de points de grille associé aux *sparse grids* de type *Maximum Norm* et *No Boundary* est plus important, et la précision pour approcher des fonctions non linéaires est meilleure en utilisant les fonctions de base de type polynômes : dans notre étude, nous utiliserons donc les types Clenshaw-Curtis et Chebyshev.

3.3 Mise en oeuvre

3.3.1 Propriété clé : qualité de parcimonie

Les propriétés suivantes découlent de la construction de Smolyak et font apparaître l'idée fondamentale des *sparse grids* : elles possèdent la qualité de parcimonie nécessaire (voir section 1.2.5 page 187) pour un travail en grande dimension.

Approche hiérarchique La formulation précédente (V.80) (voir page 227) permet d'écrire l'interpolation de rang n en fonction des interpolations de rang inférieur

$$g_n = g_{n-1} + \Delta g_n. \tag{V.82}$$

Le surplus noté Δg_n est calculé sur les nouveaux noeuds apportés par le rang n

$$\begin{aligned} \Delta g_n &:= \sum_{|\mathbf{i}|_1=d+n} (\Delta_{i_1} \otimes \ldots \otimes \Delta_{i_d}) \\ &= \sum_{|\mathbf{i}|_1=d+n} \sum_{\mathbf{j}} a_{\mathbf{j}}^{\mathbf{i}} \,.\, w_{\mathbf{j}}^{\mathbf{i}} \end{aligned} \tag{V.83}$$

avec

- $\mathbf{j} = (j_1, \ldots, j_d)$ tel que $\forall k \in \{1, \ldots, d\} \quad j_k = 1, \ldots, m_{i_k}^{\Delta}$,
- $a_{\mathbf{j}}^{\mathbf{i}} := a_{j_1}^{i_1} \otimes \ldots \otimes a_{j_d}^{i_d}$,
- $w_{\mathbf{j}}^{\mathbf{i}} := G(x_{j_1}^{i_1}, \ldots, x_{j_d}^{i_d}) - g_{n-1}(x_{j_1}^{i_1}, \ldots, x_{j_d}^{i_d})$.

On peut construire ainsi l'approximation à partir du rang $n = 0$ en rajoutant les surplus jusqu'à atteindre le rang désiré

$$g_n = g_0 + \Delta g_{d+1} + \ldots + \Delta g_{d+n}. \tag{V.84}$$

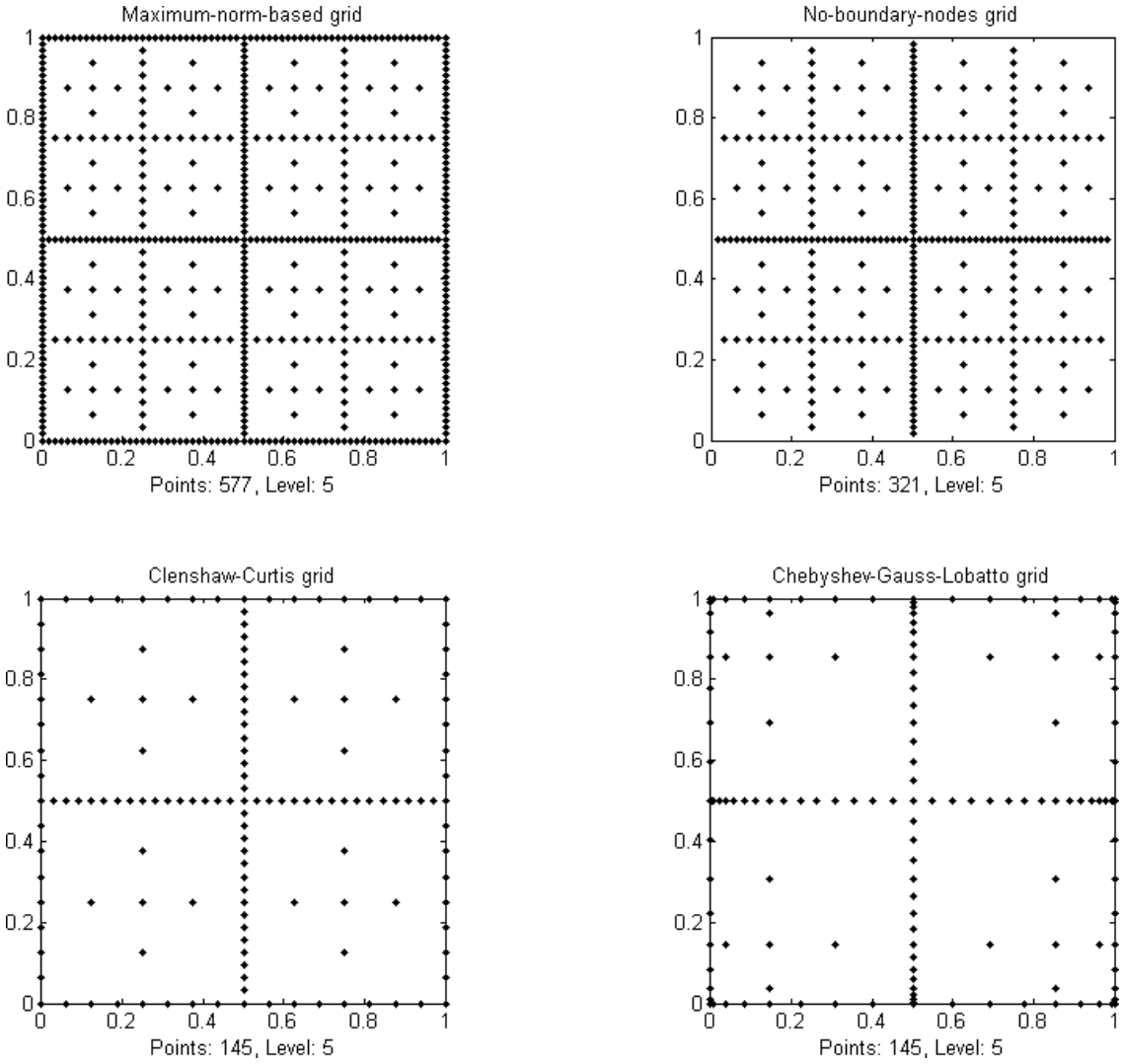


FIG. V.17 – Quatre types de *sparse grid*

Sparse grid En ce qui concerne les points de grille, on obtient la définition suivante de la *sparse grid* de rang n notée H_n comme l'ensemble des noeuds de $\mathbb{R}^d$ avec lesquels on construit g_n ; d'après (V.80) (voir page 227) et (V.82), on a

$$\begin{aligned} H_n &:= \bigcup_{|\mathbf{i}|_1 \leqslant d+n} (X_\Delta^{i_1} \times \ldots \times X_\Delta^{i_d}) \\ &= H_{n-1} \cup \Delta H_n \end{aligned} \tag{V.85}$$

avec

- $X_\Delta^i := X^i \backslash X^{i-1}$,
- $\Delta H_n := \cup_{|\mathbf{i}|_1 = d+n} (X_\Delta^{i_1} \times \ldots \times X_\Delta^{i_d})$.

Ainsi la *sparse grid* est construite de façon hiérarchique : pour atteindre une précision donnée, on augmente le rang de la *sparse grid* en gardant les données précédentes, c'est-à-dire les évaluations de la fonction G sur les points des grilles de rang inférieur.

Une grille de rang donné est incluse dans une grille de rang supérieur (voir figure V.18 avec la comparaison de grilles de rang 3 et 4 en dimension deux).

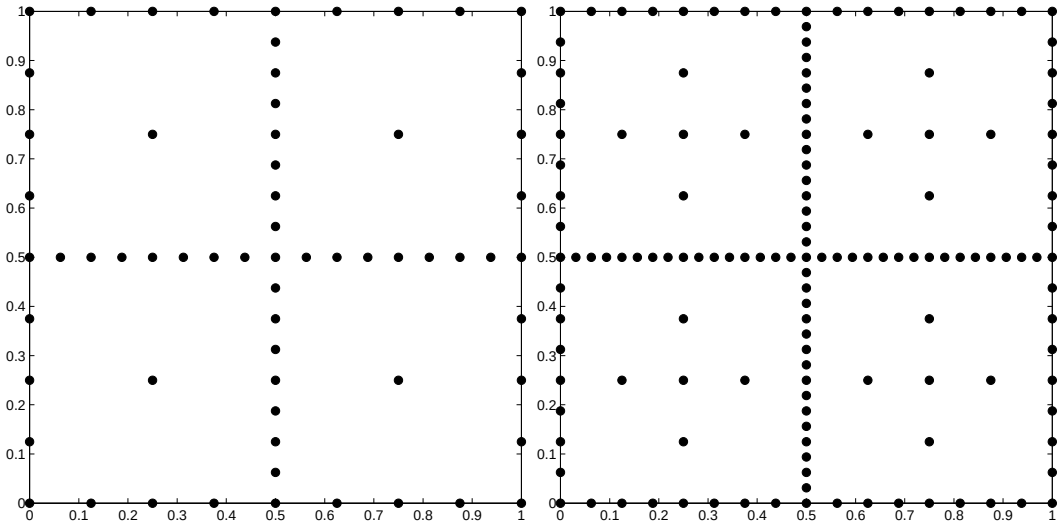


FIG. V.18 – Comparaison des grilles de rang 4 et 5 (type Clenshaw-Curtis - dimension 2)

Nombre de points de grille La méthode *sparse grid* permet de travailler avec une grille réduite, dont le nombre de points (ou nombre de fonctions de base), est très inférieur à celui d'une grille complète pour l'interpolation classique.

Grilles régulières Soit N le nombre de noeuds, et $h_N := 2^{-(N-1)}$ la distance entre deux noeuds, dans une direction.

Nous considérons ici une grille régulière, avec la même résolution dans toutes les directions, c'est-à-dire avec les notations de la section 3.2.1 (voir page 221)

$$\forall k = 1, \ldots, d \quad m_{i_k} = N. \tag{V.86}$$

Nous apellerons N ou h_N la résolution de la grille.

Pour comparer une grille pleine et une *sparse grid* de type Clenshaw-Curtis,

- Le tableau V.3 montre la comparaison du nombre de points de grille, pour une résolution fixée à $N = 5$ et quelques dimensions différentes ($d = 2, 10, 100$),

  La colonne $d = 100$ montre qu'en grande dimension le nombre de points de la *sparse grid* est nettement inférieur.

- Le tableau V.4 donne le nombre de points de grille selon la résolution N.

  Le nombre de points de la *sparse grid* augmente de façon beaucoup moins rapide en fonction de la dimension d, car le facteur élevé à la puissance d ou $d-1$ est $\log(N)$ au lieu de N.

Résolution $N = 5$	Méthode	Dimension		
		d =2	$d = 10$	$d = 100$
Nombre de points de grille	Grille pleine	$5^2 = 25$	$5^{10} \simeq 10^7$	$5^{100} \simeq 7.\ 10^9$
	Sparse grid	13	221	20201

TAB. V.3 – Nombre de points selon la dimension d ($N = 5$) : grille complète et *sparse grid* (type Clenshaw-Curtis)

Résolution N	Nombre de points de grille	Erreur $\|G - g\|_2$
Grille pleine	N^d	$\mathcal{O}(N^{-2})$
Sparse grid	$\mathcal{O}(N\ \log(N)^{d-1})$	$\mathcal{O}(N^{-2}\ \log(N)^{d-1})$

TAB. V.4 – Nombre de points et erreur d'interpolation : grille complète et *sparse grid* (type Clenshaw-Curtis)

3.3.2 Erreur d'interpolation

Avec les notations précédentes, nous présentons l'erreur d'interpolation (norme L_2 de $G-g$) en fonction de la résolution pour une interpolation classique (grille pleine) et la méthode *sparse grid* dans le tableau V.4 (voir [BG04]) : la fonction G est supposée suffisamment régulière, dans ce cas G appartient à l'espace de Sobolev H^2 (norme des dérivées secondes bornée).

La décroissance asymptotique de l'erreur quadratique de l'interpolation sur une grille pleine est conservée à un facteur logarithmique près pour les *sparse grids*.

Erreur estimée Les surplus hiérarchiques corrigent à chaque rang n la *sparse grid* de rang précédent g_{n-1} et permettent de mettre à jour l'approximation : pour des fonctions régulières, ils tendent vers zéro quand n tend vers l'infini.

La mesure des valeurs des surplus hiérarchiques constituent donc un critère naturel pour estimer l'erreur courante et contrôler l'algorithme.

On définit ainsi respectivement l'erreur estimée absolue e^n_{abs} et l'erreur estimée relative e^n_{rel} par

$$e^n_{abs} := \max_{|\mathbf{i}|_1 = d+n\ ,\ \mathbf{j}} \|w^{\mathbf{i}}_{\mathbf{j}}\|$$
$$e^n_{rel} := \frac{e^n_{abs}}{M - m}$$

avec

- $M := \max_{|\mathbf{i}|_1 \leqslant d+n,\ \mathbf{j}} G(x^{\mathbf{i}}_{\mathbf{j}})$,
- $m := \min_{|\mathbf{i}|_1 \leqslant d+n,\ \mathbf{j}} G(x^{\mathbf{i}}_{\mathbf{j}})$.

3.3.3 Propriété clé : algorithme adaptatif

Qualité de parcimonie Quand la dimension du problème devient grande (une centaine de paramètres), le nombre de points de la *sparse grid* augmente moins vite que pour une grille pleine, mais on se retrouve quand même confronté à la limitation du coût de calcul pour interpoler une fonction de modélisation.

De plus, l'expérience de l'étude des fonctions en grande dimension montre qu'en pratique, seulement une partie des directions est prépondérante et explique le comportement de la fonction : dans ce cas, il est raisonnable de concentrer le coût de calcul dans ces directions. Le problème est qu'en pratique on connaît rarement ces directions, et il faut effectuer une étude de sensibilité préliminaire afin d'éliminer certaines directions.

Pour résoudre ces difficulté, un mode adaptatif a été proposé (voir [GG03]) et implémenté dans le code de Klimke :

- l'algorithme **détecte automatiquement les directions difficiles à approcher** pendant le processus de construction de la *sparse grid* et **raffine**, c'est-à-dire continue à placer des points de grille, **dans ces directions**,
- l'algorithme arrête de placer des points dans les autres directions, qui sont faciles à apprendre.

On obtient alors à la fin de la construction de la *sparse grid*

- des niveaux élevés pour les directions importantes,
- des niveaux peu élevés pour les directions négligeables.

La qualité de parcimonie des *sparse grids* est ainsi nettement améliorée grâce au mode adaptatif pour l'approximation en grande dimension.

Principe Le principe du mode adaptatif (voir [GG03] et [Kli05]) est de choisir à chaque itération de l'algorithme un vecteur d'indices contenant les indices précédents de manière à effectuer la plus grande réduction possible de l'erreur d'approximation.

Notations Soit $S \in \mathbb{N}^d$ un ensemble de vecteurs d'indices, S est dit" admissible" si

$$\forall \mathbf{i} \in S \; \forall 1 \leqslant j \leqslant d \quad \mathbf{i} - \mathbf{e_j} \in S \tag{V.87}$$

avec $\mathbf{e_j} = (0, \ldots, 1, \ldots, 0)^T$ le j^{ieme} vecteur unité.

L'ensemble des vecteurs d'indices dits "voisins" de $\mathbf{i}$ et noté $\mathcal{V}_\mathbf{i}$ est

$$\mathcal{V}_\mathbf{i} := \{\mathbf{i} + \mathbf{e_j}; j = 1, \ldots, d\}. \tag{V.88}$$

Erreur L'erreur notée $e_{\mathbf{i}}$ et associée à un vecteur d'indices $\mathbf{i} := (i_1, \ldots, i_d)$ est

$$e_{\mathbf{i}} := \frac{1}{n_{\mathbf{i}}} \sum_{\mathbf{j} \in I_{\mathbf{i}}^{\Delta}} \|w_{\mathbf{j}}^{\mathbf{i}}\| \tag{V.89}$$

avec

- $w_{\mathbf{j}}^{\mathbf{i}}$ l'ensemble des surplus hiérarchiques de la sous-grille $\mathbf{X_i} := (X_{\Delta}^{i_1}, \ldots, X_{\Delta}^{i_d})$,
- $n_{\mathbf{i}} := card(X_i)$,
- $I_{\mathbf{i}}^{\Delta} := \{\mathbf{j} := (j_1, \ldots, j_d);\ j_k = 1, \ldots, m_{i_k}^{\Delta}\}$.

Comme l'erreur estimée de la *sparse grid*, l'erreur $e_{\mathbf{i}}$ est basé sur les surplus hiérarchiques w_j^i.

Algorithme On initialise l'algorithme adaptatif avec l'ensemble de vecteurs d'indices $S_1 := \{(1, \ldots, 1)\}$

A l'itération k avec l'ensemble de vecteurs d'indices S_k définissant la *sparse grid* courante, on ajoute tous les indices voisins qui définissent l'ensemble noté $\mathcal{V}_{S_k}$, et on calcule les erreurs associées aux nouveaux indices.

Par exemple $\mathcal{V}_{S_1} := \{(2, 1, \ldots, 1), (1, 2, \ldots, 1), \ldots (1, \ldots, 1, 2)\}$.

On choisit alors le vecteur d'indices $\mathbf{i}_k \in \mathcal{V}_{S_k}$ pour lequel l'erreur $e_{\mathbf{i}_k}$ est maximale, et on obtient la *sparse grid* à l'itération $k+1$ associée à $S_{k+1} := S_k \cup \mathbf{i}_k$.

On continue le processus jusqu'à ce que l'erreur estimée globale de la *sparse grid* soit inférieure à une tolérance donnée en entrée, ou jusqu'à ce qu'un nombre maximum de points de grille donné en entrée soit atteint.

Chapitre VI

Applications numériques

Sommaire

1 Présentation des cas tests

1.1 Centre d'Etudes de Gramat

1.1.1 Présentation

Le **CEG** (pour Centre d'Etudes de Gramat) est un centre de recherche de la DGA (pour Délégation Générale pour l'Armement) situé dans le Lot, dédié à l'analyse de la

vulnérabilité ou la protection de systèmes complets (bâtiments, véhicules, avions,...).

Le centre utilise des moyens expérimentaux et numériques importants pour la simulation des effets des armements, et a développé une expertise dans la modélisation numérique.

Collaboration La collaboration entre le CEG et le laboratoire **MIP** (pour Mathématiques pour l'Industrie et la Physique) ,de l'IMT (pour Institut de Mathématiques de Toulouse), porte sur l'étude des conséquences d'un accident de type chimique.

Nous supposons que survient un accident faisant intervenir une substance toxique qui se répand dans l'atmosphère. Une population est présente au voisinage de l'accident, et selon la manière dont le nuage toxique se disperse, il peut y avoir des victimes.

La prédiction du nombre de victimes, ainsi que la fiabilité de cette prédiction, est d'une importance cruciale pour les décisions à prendre de la part des autorités qui doivent gérer cette situation d'urgence (sécurité civile, armée,...).

L'objectif à long terme est

- construire un modèle le plus précis possible de la dispersion du produit (ce n'est pas le but de cette étude),
- disposer d'un logiciel d'aide à la décision qui analyse les conséquences de l'accident : c'est la partie qui concerne ce travail.

  Comme outil d'aide à la décision, la méthode devra être le plus automatisée possible.

  On peut envisager deux utilisations :

  - en prévention : des scénarios d'accident sont étudiés à l'avance pour établir des règles de sécurité,
  - en urgence : si un accident réel survient, on peut
    - soit utiliser un scénario déjà analysé se rapprochant des conditions de l'accident,
    - soit analyser les conséquences à l'aide du logiciel avec une contrainte de temps de calcul très forte car on veut obtenir des résultats très rapidement.

Les ingénieurs du CEG sont chargés de concevoir un outil permettant de répondre à des questions que se posent les décideurs qui commandent l'analyse de l'accident. Le laboratoire MIP apporte son expertise en méthodes mathématiques qui permettent de quantifier la réponse à ces questions et de présenter des résultats numériques qui décrivent les conséquences.

Nous présentons à présent les questions à résoudre pour faire le lien avec les problèmes rencontrés classiquement en calcul de fiabilité, et définis en section 2.2 du premier chapitre (voir page 156) :

- En mode prévention :

- **Quels sont les paramètres qui font varier le plus le nombre de victimes final ?**

  C'est le problème (P_3) des paramètres influents, qui nécessite une analyse de sensibilité.

- **Quelles sont les conditions pour lesquelles le nombre de victimes dépasse un certain seuil ?**

  Cela correspond au problème (P_2) de recherche des points de conception : les conditions sont définies par les valeurs des paramètres appartenant au domaine de défaillance, et le point de conception correspond aux conditions les plus probables.

- En mode urgence :

  - **Quelle est la probabilité d'atteindre un nombre de victimes supérieur à un certain seuil ?**

    C'est le problème (P_1) de calcul de probabilité de défaillance.

1.2 Modélisation d'un accident chimique

Nous présentons à présent la **fonction de modélisation** G sur laquelle nous effectuons l'analyse d'incertitude, et précisons les hypothèses et objectifs du problème.

Nous disposons d'un modèle simplifié de l'accident chimique sous forme d'une chaîne de logiciels

- un **logiciel de dispersion** permet de calculer, en fonction de paramètres définissant les conditions du rejet, le taux du produit au sol,
- un **logiciel de conséquences** calcule en fonction du taux précédent le nombre de victimes de l'accident.

1.2.1 Logiciel de dispersion SCIPUFF

Le logiciel **SCIPUFF** est développé par la société TITAN (Etats-Unis, voir [SPH+98]) : il calcule la dispersion d'un produit après un rejet.

Nous avons utilisé la version mise à disposition du public sur le site de la société sous forme d'exécutable (pas d'accès au code ni aux équations de modélisation) : c'est donc une modélisation de type boîte noire. La version ne comporte pas toutes les fonctionnalités du logiciel commercial mais le modèle proposé nous permet néanmoins de tester nos méthodes.

Entrées Le logiciel prend en entrée de nombreuses paramètres qui déterminent les conditions de la dispersion et sont normalement choisis à l'aide d'une interface graphique :

- paramètres concernant la nature et les propriétés du produit (taille des gouttes,...),
- paramètres de rejet (coordonnées, durée, hauteur,..),

- paramètres définissant les conditions naturelles :
  - météorologie (présence nuageuse, de pluie, de vent,...),
  - environnement(présence de végétation,...).
- paramètres de calcul (pas de discrétisation, temps de calcul,...).

Sortie Le logiciel donne en sortie plusieurs résultats sur la dispersion du produit, mais celui qui nous concerne est le taux du produit au sol, calculé sur une grille de discrétisation.

1.2.2 Logiciel de conséquences

Ce logiciel a été développé en fortran au CEG et est aussi fourni sous forme d'exécutable.

Il modélise de manière très simple des lieux d'habitation par trois rectangles avec des densités d'habitation différente et utilise un modèle de létalité simplifié afin de calculer le nombre de victimes en fonction du taux de produit dans la zone de population (voir figure VI.1).

Entrée Le logiciel prend en entrée la sortie du logiciel de dispersion, c'est-à-dire le taux du produit dispersé.

Sortie Le logiciel donne en sortie un nombre entier : le nombre de victimes de l'accident.

1.2.3 Interfaçage

Nous avons interfacé les deux logiciels précédents avec des appels définis dans des fichiers batchs qui permettent d'effectuer les calculs sans passer par l'interface graphique du logiciel de dispersion, condition obligatoire pour l'utilisation sous forme de fonction de modélisation.

La modification des paramètres d'entrée s'effectue en écrivant directement dans les fichiers SCIPUFF qui les incorporent.

La figure VI.1 présente une capture d'écran qui représente la dispersion du produit obtenue avec le logiciel SCIPUFF : on voit l'origine du rejet et le nuage obtenu. Nous avons ajouté les zones de population prises en compte par le logiciel de conséquences : ici le rectangle horizontal est atteint, il y aura vraisemblablement des victimes, mais si les conditions de rejet sont différentes, on voit qu'il peut n'y avoir aucune victime.

1.2.4 Fonction de modélisation

La chaîne de logiciels précédente représentée dans la figure VI.2 définit une fonction de modélisation correspondant aux hypothèses formulées quand nous avons posé le

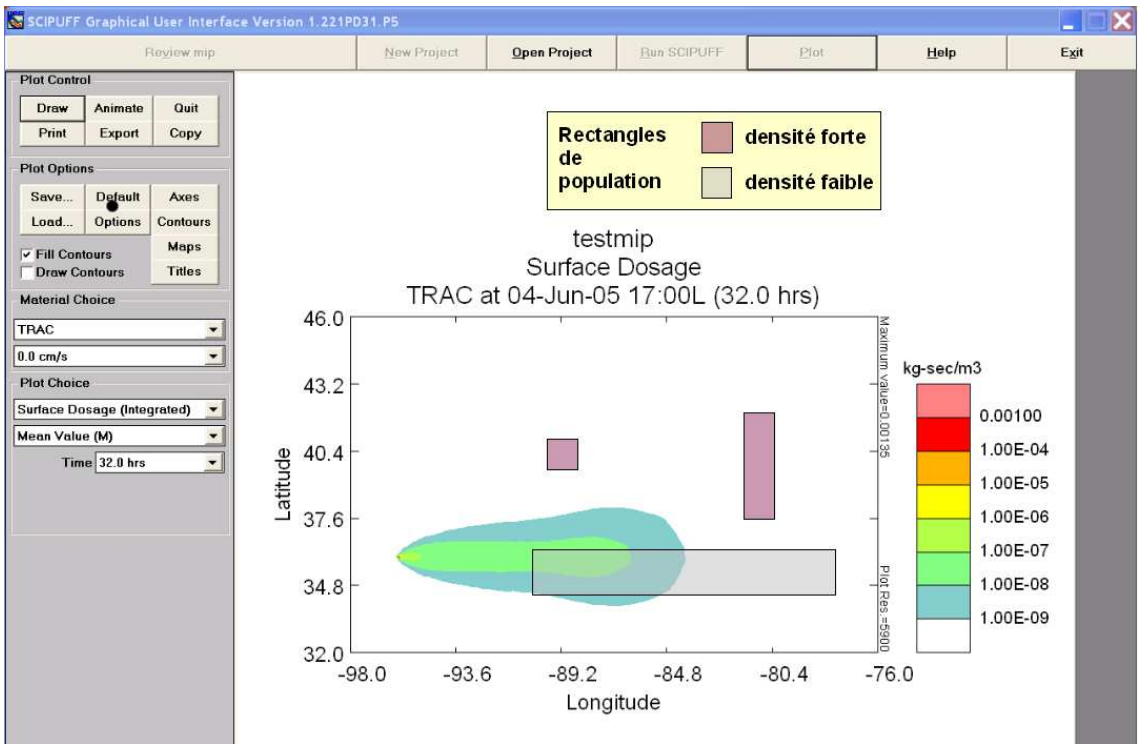


FIG. VI.1 – Aperçu du logiciel de calcul de dispersion SCIPUFF avec les zones de population

problème dans la section 2.1 du premier chapitre (voir page 152) : nous reprenons les mêmes notations et nous avons ici

$$\begin{array}{rccc} G: & \Omega \subset \mathbb{R}^n & \longrightarrow & \mathbb{R}^+ \\ & X = (x_1, \ldots, x_n) & \longmapsto & G(X) \end{array} \tag{VI.1}$$

avec

$$\Omega = \prod_{i=1}^{n} [a_i, b_i] \tag{VI.2}$$

car la plage des valeurs autorisées dans le logiciel de dispersion est bornée pour tous les paramètres.

Nous supposons que le coût d'une évaluation du modèle G est important : la partie coûteuse est le calcul de dispersion.

Dans le cadre de cette étude, le modèle simplifié fourni par SCIPUFF nous permet d'effectuer une évaluation en quelques secondes, mais la méthode doit s'appliquer à des modèles plus réalistes dont le coût sera de l'ordre de plusieurs minutes.

1.2.5 Paramètres incertains et scénarios

Lors de la modélisation d'un accident, on peut considérer que la majorité des paramètres sont incertains car on envisage un événement suceptible de se produire dans l'avenir : on ne connaît pas à l'avance les caractéristiques exactes du rejet et des conditions naturelles.

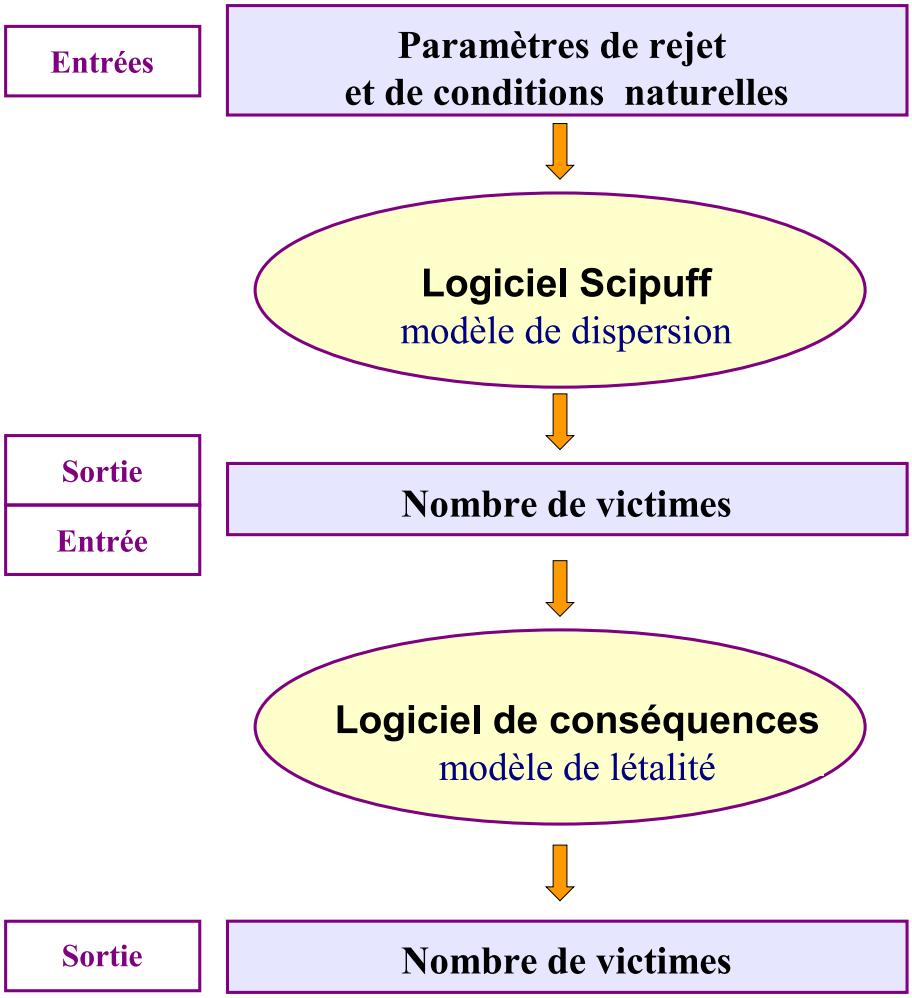


FIG. VI.2 – Chaîne de logiciels

Ces paramètres seront définis à l'aide d'une loi de probabilité : les hypothèses sur la distribution peuvent provenir de données disponibles (par exemple des relevés météorologiques dans un lieu donné) ou de la connaissance d'un expert.

On peut d'autre part décider de fixer des paramètres à des valeurs constantes pour pouvoir tester des configurations précises : le rejet d'un produit donné, une explosion située exactement à tel endroit,...

Le choix des paramètres fixes et des distributions de probabilité des paramètres incertains est effectué par le CEG qui définit ainsi un scénario d'accident à évaluer.

Lois de probabilité Soit le vecteur des paramètres incertains $\mathbf{X} = (\mathbf{x_1}, \ldots, \mathbf{x_n})$, dans cette étude nous supposerons chaque $\mathbf{x_j}$ est une variable aléatoire régie

- soit par une **loi uniforme**,
- soit par une **loi gaussienne** de moyenne m_j et d'écat-type σ_j, notée $N(m_j, \sigma_j)$.

Nous notons X_j une réalisation de $\mathbf{X}$.

Nous cherchons à connaître les caractéristiques de la variable aléatoire $G(\mathbf{X})$.

Remarque Nous avons vu précédemment que les paramètres sont définis sur des intervalles bornés, or normalement les lois gaussiennes sont définies sur $\mathbb{R}$.

Dans l'implémentation, suivant la valeur des écarts-types, les lois gaussiennes sont donc tronquées : pour un tirage aléatoire de taille N à réaliser, les points hors de l'intervalle de définition ne sont pas pris en compte, et on effectue de nouveaux tirages jusqu'à obtenir N points dans l'intervalle.

Nous notons

- I_U les indices des lois uniformes,
- I_N les indices des lois gaussiennes.

2 Méthode par approximation globale

2.1 Rappel des problèmes à résoudre et objectif

Nous rappelons brièvement les problèmes à résoudre, déjà introduits en section 2.1, et l'objectif poursuivi.

2.1.1 (P_1) Probabilité de défaillance p_s

Pour un seuil donné $s \in \mathbb{R}^+$, nous cherchons la **probabilité de dépasser ce seuil**

$$\begin{array}{lll} (P_1) & p_s & := p(s \leqslant G(\mathbf{X})) \\ & & = p(G(\mathbf{X}) \in D_s) \\ & & = \int_{D_s} \rho(\mathbf{X}) d\mathbf{X} \\ & & = \int_{D_s} \rho(\mathbf{x_1}) \dots \rho(\mathbf{x_n}) d\mathbf{x_1} \dots d\mathbf{x_n} \end{array} \tag{VI.3}$$

avec $D_s = \{X \in \mathbb{R}^n;\ s \leqslant G(X)\}$ le **domaine de défaillance**.

2.1.2 (P_2) Points de conception

Avec le changement de variable $\mathbf{y_i} = \frac{\mathbf{x_i} - m_i}{\sigma_i}$ pour $i \in I_N$, les paramètres de loi gaussienne suivent une loi centrée réduite $N(0,1)$.

La recherche des **points du domaine de défaillance les plus probables**, dits points de conception P^*, s'écrit sous la forme du problème d'optimisation sous contrainte

$$\begin{array}{lll} (P_2) & P^* & = \min_{X \in D_s} \sum_{i \in I_N} y_i^2 \\ & & = \min_{X \in D_s} \sum_{i \in I_N} \left(\frac{x_i - m_i}{\sigma_i} \right)^2 . \end{array} \tag{VI.4}$$

2.1.3 (P_3) Paramètres influents

Nous cherchons un moyen de **quantifier l'influence des paramètres sur la réponse $G(X)$ pour déterminer les plus importants**. L'idée est d'utiliser éventuellement cette information pour **réduire la taille du problème** : on effectue l'étude de G uniquement pour ces paramètres influents, en fixant la valeur des autres paramètres.

Nous considérons un paramètre x_j comme influent, ou important, si les variations de x_j sur son intervalle de définition $[a_j, b_j]$ entraînent des variations significatives sur la réponse $G(X)$.

2.1.4 Objectif : coût de calcul

Nous voulons effectuer une étude de fiabilité, consistant à résoudre les trois problèmes (P_i) précédents, pour une fonction de modélisation G supposée coûteuse, et en vue d'une application possible en grande dimension ($n \simeq 100$).

Quelles que soient la méthode d'approximation, puis les méthodes de résolution des problèmes (P_i) choisies, nous considérons que le coût de calcul lié aux appels à la fonction G est prépondérant.

Un critère important pour évaluer les résultats sera donc le nombre d'évaluations de la fonction G, c'est-à-dire le nombre de données nécessaires à la construction de l'approximation g.

Dans les cas tests présentés, le modèle utilisé est simple et donc peu coûteux (une évaluation de l'ordre de quelques secondes), mais il faudra obtenir des résultats satisfaisants avec le moins de simulations possible.

2.2 Principe général

Les étapes de la méthode que nous appliquons sont :

- **Construire une approximation** g de G, deux types d'approximation sont retenues
  - par **réseau de neurones**,
  - par interpolation ***sparse grid***.

  Les principes de ces deux méthodes ont été présentés en section 2 (voir page 205) et en section 3 (voir page 220) du chapitre précédent, et nous discuterons leur mise en oeuvre dans les sections 3 (voir page 258) et 4 (voir page 276).

  Cette étape est cruciale, car de la qualité de l'approximation g vont dépendre les résultats de l'étude de G.

  Les propriétés les plus importantes de l'approximation g sont
  - un coût d'évaluation négligeable,
  - la possibilité d'utiliser le gradient.
- **Utiliser g pour résoudre les problèmes**

- (P_1) calcul des **probabilités de dépasser un seuil**,
- (P_2) calcul des **points de conception** (optimisation sous contraintes),
- (P_3) **analyse de sensibilité**.

2.3 Approximation par réseau de neurones

2.3.1 Caractéristiques de l'apprentissage

La méthode de réseau de neurones présente une phase délicate, celle de l'apprentissage.

En effet, selon le plan d'expériences choisi pour les données (nombre et disposition) et les valeurs des paramètres de l'algorithme (critères d'arrêt, nombre de neurones cachés, forme de régularisation,...), on obtient une approximation de qualité différente.

Paramètres de l'algorithme Dans notre cas, le nombre de neurones cachés augmente automatiquement jusqu'à ce que le réseau neuronal atteigne une certaine précision, et les paramètres de l'algorithme les plus sensibles sont

- le nombre de données noté N,
- la tolérance notée τ : elle définit la précision souhaitée.

  Dans l'algorithme, elle est utilisée sous la forme suivante :

  - soit le lot noté $\{X_1, \ldots, X_{N_1}\}$ des données pour la phase dite de l'apprentissage par coeur,
  - on pose $\tau_2 := \tau * V(C)$ avec $V(C)$ la variance empirique du vecteur des cibles $C = (G(X_1), \ldots, G(X_1)^T$
  - soit $r := \frac{1}{N_1} \sum_{i=1}^{N_1} (G(X_i) - g(X_i))^2$ l'erreur quadratique moyenne au carré,
  - le critère d'arrêt est alors $r < \tau_2$.

Exemples simples Nous présentons l'influence des deux paramètres de l'algorithme précédent sur des exemples de fonctions tests en dimension deux.

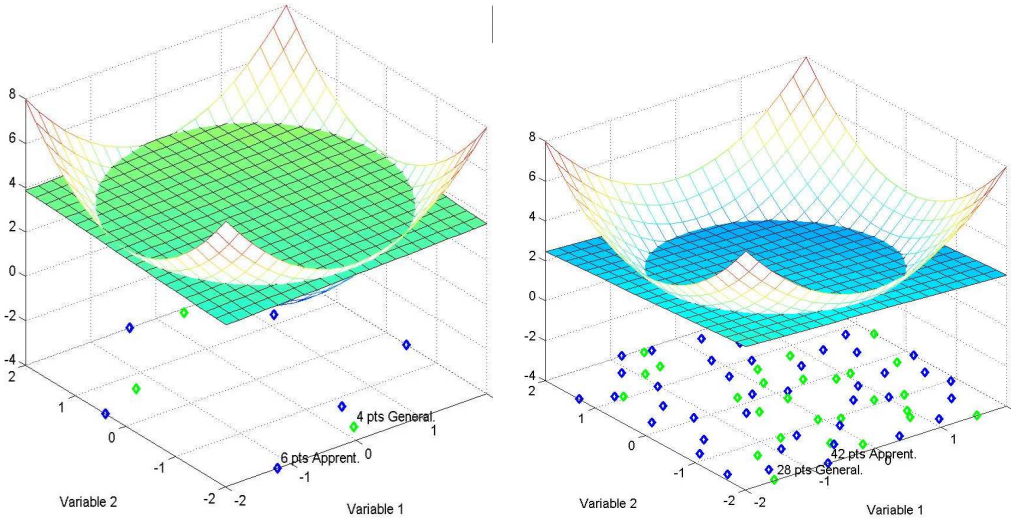
Dans les figures suivantes, les données d'apprentissage sont marquées dans le domaine de départ (plan horizontal), la fonction est représentée par une surface transparente et l'approximation du réseau de neurones est représentée par une surface pleine.

Nous utilisons un nombre $N = N_1 + N_2$ de données d'apprentissage : N_1 pour la phase d'apprentissage par coeur et N_2 pour la phase de régularisation.

Tolérance grande Si la tolérance τ est grande, le réseau neuronal ne va pas chercher à passer exactement par les cibles, et il va devenir une approximation simplifiée de la fonction.

Cette propriété est illustrée avec la figure VI.3, avec $\tau = 5\ 10^{-}1$ et la fonction carré :

- avec $N = 10$, le réseau neuronal obtenu est une constante, égale à la moyenne des valeurs apprises (figure VI.3(a)).
- Si on augmente le nombre de données, avec $N = 70$, le réseau neuronal reste une approximation d'ordre zéro (constante), qui converge vers l'espérance de la fonction carré sur le domaine considéré $[-2, 2]^2$ (figure VI.3(b)).



(a) N=10 (b) N=70

FIG. VI.3 – Réseau neuronal avec $\tau = 0.5$ pour la fonction carré

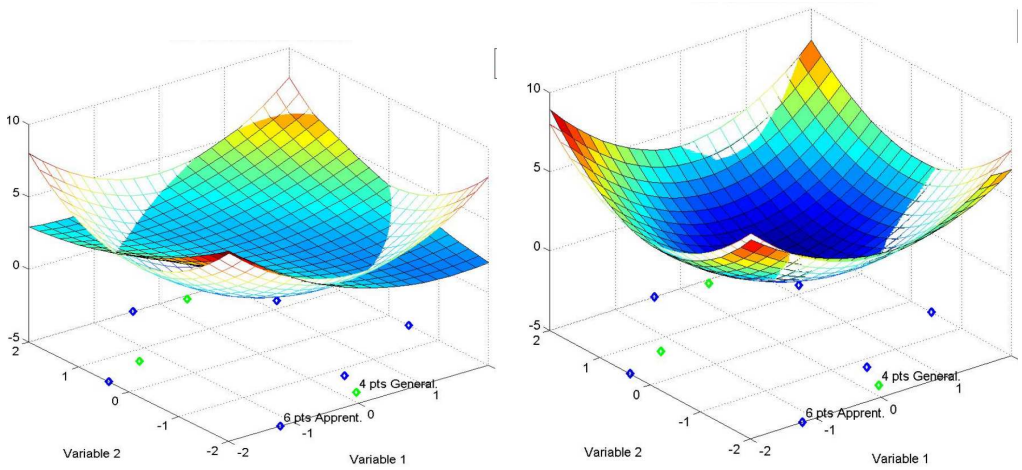
Tolérance petite Si on prend une tolérance plus petite, le réseau neuronal sera plus proche de la fonction.

Sur la figure VI.4, avec la fonction carré :

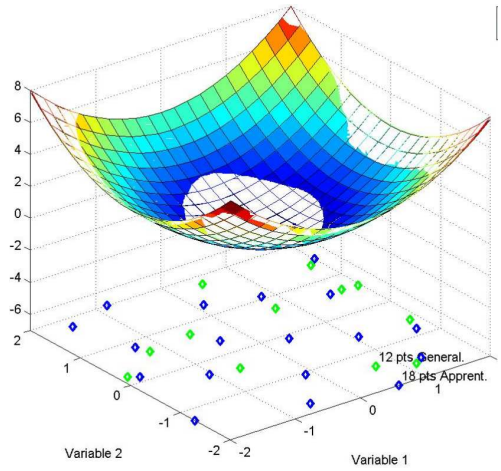
- avec $N = 10$ et $\tau = 5\ 10^{-}2$, le réseau neuronal devient non linéaire, mais l'approximation n'est pas satisfaisante (figure VI.4(a)).

  On obtient une bonne approximation en trouvant le compromis entre :

  - augmenter le nombre de données,
  - et diminuer la tolérance.
- Si on diminue la tolérance à données fixées, avec $N = 10$ et $\tau = 5\ 10^{-}3$, l'approximation est à présent satisfaisante (figure VI.4(b)).
- Si on augmente le nombre de données avec une tolérance plus grande, avec $N = 30$ et $\tau = 10^{-}1$, on obtient la même qualité (figure VI.4(c)).



(a) N=10 et $\tau = 10^-2$ (b) N=10 et $\tau = 10^-3$



(c) N=30 et $\tau = 10^-1$

FIG. VI.4 – Réseau neuronal avec différentes valeurs de N et τ pour la fonction carré

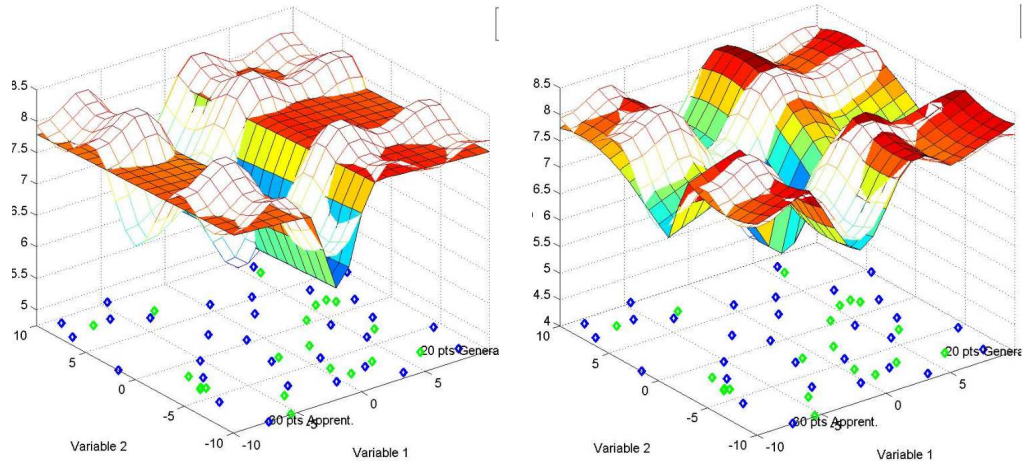
Conclusion Plus le nombre de points d'apprentissage est grand et plus la tolérance est petite, de meilleure précision sera l'approximation obtenue.

Pour des fonctions plus difficiles à apprendre, c'est-à-dire non régulières, il faut choisir un nombre assez grand de points d'apprentissage, sinon le réseau neuronal n'arrive pas à apprendre malgré une tolérance petite.

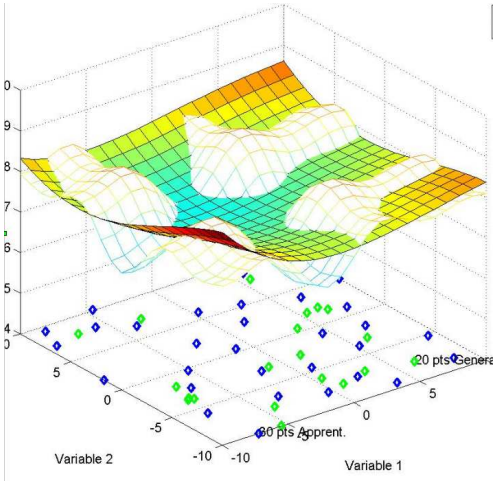
Mais si on prend un grand nombre de données, il ne faut pas choisir une tolérance trop petite, sinon le réseau neuronal n'arrive pas à passer précisément par tous les points et donne une mauvaise approximation, l'algorithme est mis en échec.

Cette propriété est illustrée sur la figure VI.5 avec la fonction sinus décalé en dimension deux. Avec $N = 50$, on obtient

- avec $\tau = 5\ .\ 10^-1$ (figure VI.5(a)) et $\tau = 10^-1$ (figure VI.5(b)), la précision du réseau neuronal augmente,
- avec $\tau = 10^-3$, la précision est très dégradée.



(a) $\tau = 5\ .\ 10^{-}1$ (b) $\tau = 10^{-}1$



(c) $\tau = 10^{-}3$

FIG. VI.5 – Réseau neuronal avec différentes valeurs de N et τ pour la fonction carré

2.3.2 Validation du réseau neuronal

Nous présentons ici l'affichage qui permet de vérifier la qualité d'un réseau neuronal.

Pour valider l'approximation obtenue, nous considérons trois lots de données :

- un lot $(X_1, \ldots, X_N)$ généré par deux hypercubes latins pour la phase d'apprentissage (un pour le lot d'apprentissage par coeur et l'autre pour le lot de régularisation),
- un lot $(X_1, \ldots, X_{N_v})$ généré par un tirage de type carré latin pour la phase de vérification.

Graphique de validation Les cibles et les valeurs correspondantes sont représentées pour l'ensemble des données (lots d'apprentissage par coeur, de régularisation, et de vérification) sur le graphique de validation par les points $(G(X_i), G(X_i))$ et $(G(X_i), g(X_i))$.

Les cibles sont donc disposées sur la droite $y = x$ et on visualise l'écart avec l'approximation sur la même abscisse.

Coefficient de corrélation En considérant g et G comme deux variables aléatoires dont les réalisations sur un lot de données sont les vecteurs $\widehat{g} = (g(X_1), \ldots, g(X_N))$ et $\widehat{G} = (G(X_1), \ldots, G(X_N))$, on peut calculer le coefficient de corrélation

$$Cor = \frac{cov(\widehat{g}, \widehat{G})}{\sigma_{\widehat{g}}\ \sigma_{\widehat{G}}} \tag{VI.5}$$

avec *cov* la covariance et σ l'écart-type des vecteurs considérés.

Ce coefficient, compris dans notre cas entre 0 et 1 mesure la dépendance linéaire entre $\widehat{g}$ et $\widehat{G}$: plus il est proche de 1, plus la dépendance linéaire entre les deux vecteurs est forte, ce qui normalement signifie que les valeurs $g(X_i)$ sont proche de la droite $y = x$ et donc des cibles $G(X_i)$.

Il est calculé sur le lot d'apprentissage, noté $CorA$, puis sur le lot de vérification, noté $CorV$.

Erreur quadratique moyenne L'erreur quadratique moyenne relative, notée EQM, sur un lot $\{X_1, \ldots, X_N\}$ est calculée par

$$EQM := \frac{1}{(M_G - m_G)} \sqrt{\frac{1}{N} \sum_{i=1}^{N} (g(X_i) - G(X_i))^2} \tag{VI.6}$$

avec

- $M_G = \max_{i=1\ldots N} G(X_i)$,
- $m_G = \min_{i=1\ldots N} G(X_i)$.

Elle est calculée sur le lot d'apprentissage et notée $EQMA$ puis sur le lot de vérification et notée $EQMV$.

Pour valider un réseau neuronal, le principe est d'avoir un coefficient de corrélation Cor le plus proche de un possible et une erreur EQM la plus proche de zéro possible.

En pratique, pour sélectionner un réseau neuronal approchant une fonction donnée, on vérifie qu'on obtient des ordres de grandeur équivalents sur les lots d'apprentissage et de vérification pour Cor et EQM, puis on compare les valeurs de Cor et EQM avec celles d'autres réseaux neuronaux pour dégager le meilleur.

2.3.3 Sigmoïde en sortie

La fonction de modélisation présente la caractéristique de donner une réponse égale à zéro sur une grande partie du domaine de définition.

Du point de vue du phénomène modélisé, cela est dû au fait que les conditions pour que le nuage toxique parvienne sur les zones de population ne sont réunies que pour certaines valeurs des paramètres d'entrée.

	Lot d'apprentissage	Neurones cachés	$EQMA$	$EQMV$
g_{lin}	1000	21	0.052	0.099
g_{sig}	1000	4	0.064	0.062
g_{linF}	1000	21	0.049	0.092

TAB. VI.1 – Erreurs sur lot d'apprentissage $EQMA$ et vérification $EQMV$ pour les réseaux neuronaux g_{lin}, g_{sig} et g_{linF}

Le réseau neuronal a alors tendance à extrapoler les valeurs nulles par des valeurs négatives : une idée pour améliorer le comportement du réseau neuronal est de **remplacer la fonction d'activation linéaire du neurone de sortie** par la même **fonction d'activation sigmoïde** que celle de la couche cachée.

Avec une même tolérance et un même lot de données de 1000 points, nous construisons :

- un réseau neuronal g_{lin} avec la fonction d'activation linéaire,
- un réseau neuronal g_{sig} avec la fonction sigmoïde,
- un réseau neuronal g_{linF} en appliquant un filtre $f(x) = max(x, 0)$ à la sortie de g_{lin} ; nous notons $g_{linF} = f \circ g_{lin}$ l'approximation.

Le lot de validation est de 100 points.

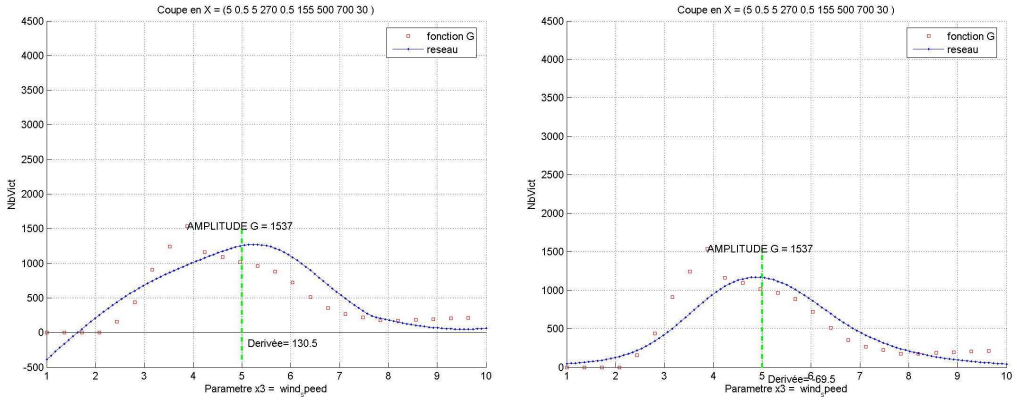
Le tableau VI.1 présente les résultats.

En comparant g_{linF} et g_{lin} : l'erreur d'apprentissage et de vérification ne diminuent que très légèrement, cette approche n'est donc pas suffisante.

En comparant g_{sig} et g_{lin} :

- L'erreur sur le lot d'apprentissage est légèrement dégradée pour g_{sig}, mais il utilise aussi beaucoup moins de neurones cachés (4 pour g_{sig} contre 21 pour g_{lin}), ce qui montre une **meilleure qualité de parcimonie**.
- Par contre, l'erreur sur le lot de vérification diminue significativement avec g_{sig}, ce qui montre une **meilleure qualité de prédiction**.

Nous voyons sur la figure VI.6 des coupes suivant un paramètre qui montrent comment le comportement du réseau neuronal est amélioré.



(a) Réseau neuronal g_{lin} (b) Réseau neuronal g_{sig}

FIG. VI.6 – Coupes des réseaux g_{lin} et g_{sig} suivant le paramètre n°3

2.4 Approximation par sparse grid

2.4.1 Caractéristiques

La méthode des *sparse grids* est basée sur une **interpolation sur grille éparse**, dont les coordonnées sont déterminées et fixées selon le type de fonction de base choisi.

Ainsi, pour une fonction à approcher sur un domaine, et pour un type de fonction de base donnés, nous obtenons une **unique approximation** *sparse grid* (contrairement à la méthode des réseaux de neurones, dont le résultat dépend de l'apprentissage).

Paramètres de l'algorithme Les paramètres sont ici plus simples à fixer que pour la méthode précédente : en effet, l'algorithme s'arrête

- si l'erreur estimée devient inférieure à une tolérance donnée en entrée,
- ou si un nombre maximum de points de grille est atteint.

Etant donné que le coût de calcul est un critère que nous voulons maîtriser, nous donnerons en entrée une tolérance assez basse pour qu'elle ne stoppe pas l'algorithme, et nous indiquerons seulement le nombre de points de grille souhaité, correspondant au nombre d'évaluations de la fonction de modélisation.

2.4.2 Validation de la sparse grid

Comme c'est une méthode d'interpolation, l'erreur sur les données de construction, constituées par les points de la grille, est égale à zéro (de l'ordre de la précision machine lors des calculs).

L'algorithme donne en sortie une **erreur relative** E_R estimée, qui permet d'apprécier la précision de l'approximation. Si nous notons w_k les surplus du dernier niveau calculé (voir section 3.2.1 page 221 du chapitre précédent), E_R est définie comme le maximum des surplus normalisé sur les données

$$E_R := \frac{\max_k |w_k|}{\max_{i=1\ldots N} G(X_i)}. \tag{VI.7}$$

Nous verrons dans les tests numériques que l'erreur E_R n'est pas suffisante pour valider la *sparse grid*.

Donc, pour comparer les approximations, nous calculons aussi l'erreur quadratique $EQMV$ sur un tirage de vérification de type carré latin (nous n'utilisons pas les lois de probabilité du scénario) commun à toutes les *sparse grids* calculées.

La validation est ici différente par rapport aux réseaux de neurones : puisque pour un coût donné nous obtenons une seule *sparse grid*, si l'erreur estimée est trop grande ou si les résultats ne sont pas satisfaisants, nous n'avons que la possibilité d'augmenter le nombre de points (si c'est possible) pour améliorer l'approximation.

2.5 Méthode retenue pour (P_1)

Pour calculer les probabilités de défaillance p_s, étant donné que le coût de calcul n'est plus un obstacle, nous utilisons une méthode de Monte-Carlo avec g pour la simplicité de sa mise en oeuvre (voir section 3.2 page 159 du premier chapitre).

2.5.1 Graphique du calcul des probabilités

Le résultat des probabilités de défaillance est représenté par un graphique que nous détaillons ici (voir par exemple la figure VI.8 page 260).

Pour **valider le résultat**, nous avons aussi utilisé la **méthode de Monte-Carlo appliquée à** G avec un tirage aléatoire de 10^4 points : c'est un calcul coûteux qui n'est possible ici que parce que le modèle utilisé est très simple (une évaluation en quelques secondes) et dont le but est uniquement de disposer de valeurs de probabilité de référence pour comparaison avec notre méthode.

Nous appliquons ensuite la méthode de Monte-Carlo à l'approximation g avec un tirage de 10^4 ou 10^5 pour points (peu coûteux).

Nous affichons les résultats suivants dans le graphique :

- en abscisses sont portés les seuils de victimes s obtenus par discrétisation de l'intervalle $[0, m_G]$, avec m_G le maximum de victimes défini par le maximum de G sur les données de construction,
- en ordonnées est portée la probabilité de dépasser chaque seuil
  - pour la fonction de modélisation G par des symboles "□", et l'estimation de l'intervalle de confiance à 95% de la méthode de Monte-Carlo est représentée par des pointillés,
  - pour l'approximation g par des symboles "+".

2.6 Méthode retenue pour (P_2)

Le problème d'optimisation sous contraintes associé à la **recherche des points de conception en utilisant l'approximation** g s'écrit (voir formule (VI.4) page 243) :

$$(\widehat{P_2}) \quad P^* = \min_{X \in \widehat{D_s}} \sum_{i \in I_N} \left(\frac{x_i - m_i}{\sigma_i} \right)^2 \tag{VI.8}$$

avec $\widehat{D_s} := \{X \in \mathbb{R}^n;\ s \leqslant g(X)\} \quad \simeq \quad D_s = \{X \in \mathbb{R}^n;\ s \leqslant G(X)\}$
le domaine "approché" de défaillance.

Pour résoudre le problème d'optimisation, nous utilisons un algorithme classique de descente, qui intègre les contraintes par pénalisation, appliqué à g dont on connaît le gradient.

Nous lançons alors l'algorithme avec un choix de points de départ aléatoires, et nous retenons les cinq premiers résultats de convergence.

2.6.1 Sortie de la recherche des points de conception

Le résultat de la recherche est donné sous la forme d'un fichier texte (voir par exemple le tableau VI.5 page 261) avec :

- le seuil s de victimes auquel est associé le domaine de défaillance D_s,
- les indices des paramètres sur lesquels on effectue l'optimisation avec les lois associées
  - la fonction coût est définie uniquement par rapport à ceux qui suivent une loi gaussienne,
  - mais le domaine des contraintes D_s est défini à partir de tous les paramètres.
- pour chaque convergence vers un point de conception P^*
  - le résidu de la fonction coût à la fin de l'algorithme (la distance à zéro OP^* dans l'espace des lois centrées réduites),
  - les coordonnées de P^*,
  - les images $g(P^*)$ et $G(P^*)$ pour vérifier si P^* est au bord du véritable domaine de défaillance D_s.

2.7 Méthode retenue pour (P_3)

Nous voulons utiliser l'avantage que nous procure le gradient de l'approximation de g pour déterminer les paramètres les plus influents.

2.7.1 Idée clé : distribution des gradients

Gradient et variations Le gradient de g évalué en un point X_k donne une information locale sur les variations de g autour du point X_k.

Nous supposons que l'approximation g est régulière et "lisse" les comportements de la fonction G.

Ce comportement a été constaté pour les réseaux de neurones grâce à l'emploi de techniques de régularisation (voir l'exemple en section 2.3.5 page 215 du chapitre précédent).

Nous supposons ainsi que si un paramètre négligeable produit des variations locales de G de type bruit et pas de variations globales significatives, alors g "élimine les variations parasites" et ne conserve que l'allure générale et la variation globale de G. Le gradient de G suivant un tel paramètre aura une valeur importante pour de nombreux points, mais le gradient de g aura une valeur faible partout.

Donc les directions dans lesquelles les variations de g sont les plus fortes localement, détectées par le gradient, indiquent que ces directions correspondent à des paramètres influents, car une fonction lisse prolonge la variation locale et on obtient les variations globales significatives de g, correspondant à celles de G si l'approximation a une bonne qualité de prédiction.

Cas des *sparse grids* Le comportement d'approximation régulière n'est par contre pas assuré pour les *sparse grids*, d'autant plus qu'avec une méthode d'interpolation, les oscillations apparaissent facilement : il faudrait leur appliquer aussi des techniques de régularisation pour obtenir une approximation lissée, qui ne serait plus une interpolation.

Pour valider la méthode de distribution des gradients avec les *sparse grids*, nous pouvons comparer le classement des paramètres influents avec celui obtenu grâce aux indices du mode adaptatif (voir section 2.7.4 page 257).

Principe Ici nous faisons intervenir les variations des paramètres sur leur intervalle de définition : c'est une mesure de sensibilité globale, avec une notion de paramètre influent qui ne dépend pas des lois de probabilité associées.

L'idée est d'**évaluer le gradient de g en de nombreux points bien répartis** dans le domaine : nous utilisons un **tirage de type carré latin** de p points (nous ne faisons donc pas intervenir les lois de probabilité).

Nous formons la matrice A avec la valeur absolue de ces gradients en colonnes.

Chaque ligne L_k de A représente les valeurs absolues des dérivées partielles de g suivant paramètre x_k évaluées sur le plan d'expériences : les paramètres influents sont ceux pour lesquels ces valeurs sont les plus fortes.

Il nous faut à présent extraire cette information de la matrice A.

Si nous prenons la norme des vecteurs lignes L_k, nous aurons un critère simple de classement de l'influence des paramètres puisque il suffit de comparer les n valeurs entre elles.

Mais nous avons choisi de **représenter la distribution** des valeurs des lignes L_k avec les **diagrammes en boîtes**, qui permettent de visualiser une information plus riche : ce diagramme permet la représentation synthétique des caractéristiques des observations avec leurs intervalles interquartiles (voir [Sap06]), afin de comparer facilement des sous-groupes de données.

2.7.2 Algorithme de recherche des paramètres importants

D'après ce qui précède, nous proposons l'algorithme suivant, qui peut être utilisé avec une approximation par réseau de neurones ou par *sparse grid*, présenté page 256.

Si nous voulons automatiser le classement des paramètres, nous pouvons choisir par exemple le troisième quartile dans la dernière étape de l'algorithme et comparer les paramètres entre eux uniquement par rapport à ce critère.

2.7.3 Graphique de recherche des paramètres importants

Le résultat de la recherche des paramètres principaux est donné par un graphique dans lequel, pour chaque paramètre x_k, sont représentées les caractérisques des valeurs des gradients dans un diagramme en boîte (ou "boîte à moustaches") vertical, dont nous détaillons le principe (voir par exemple la figure VI.9 page 263) :

Algorithme 1 : Recherche des paramètres importants

▷ **Entrées**

- $X = (x_1, \ldots, x_n)$ les paramètres d'entrée de la fonction G,
- $[a_i, b_i]\ , i = 1 \ldots n$ les intervalles de définition des x_i,
- g l'approximation de G.

▷ **Calcul des gradients**

- **Normaliser les intervalles** $[a_i, b_i]$ pour effectuer l'étude sur $[0, 1]$ pour chaque paramètre,
- Générer un plan d'expériences de type **carré latin** (sans les lois de probabilités), on obtient les points $\{X_1, \ldots, X_p\}$,
- Evaluer le gradient en ces points, on obtient $\{\nabla g(X_j)\ , i = 1 \ldots p\}$,
- Former la matrice de la valeur absolue des gradients

$$A = (|\nabla g(X_1)| \ldots |\nabla g(X_p)|) \tag{VI.9}$$

de taille $n \times p$.

▷ **Visualisation et classement des paramètres**

- **Représenter la distribution** des valeurs des n lignes

$$Lk = \left(\left| \frac{\partial g}{\partial x_k}(X_1) \right| \ldots \left| \frac{\partial g}{\partial x_k}(X_p) \right| \right) \tag{VI.10}$$

par des diagrammes en boîtes.

- Suivant les valeurs de dérivées partielles les plus fortes (par exemple en comparant les troisièmes quartiles), **déterminer un classement des paramètres** avec les catégories
  - paramètres **principaux**,
  - paramètres **secondaires**,
  - paramètres **négligeables**.

TAB. VI.2 – Algorithme de recherche des paramètres importants

- la boîte contient la moitié des valeurs :
  - la partie inférieure de la boîte contient les valeurs comprises entre le premier quartile Q_1 et la médiane (ou deuxième quartile Q_2),
  - la partie supérieure contient les valeurs comprises entre la médiane et le troisième quartile Q_3.
- les moustaches s'étendent
  - vers le bas jusqu'à la valeur la plus proche de $Q_1 - 1.5\ (Q_3 - Q1)$,
  - vers le haut jusqu'à la valeur la plus proche de $Q_3 + 1.5\ (Q_3 - Q1)$.
- les valeurs au-delà des moustaches sont repérées par des points.

2.7.4 Utilisation des *sparse grids* en mode adaptatif

L'algorithme *sparse grid* adaptatif présente un intérêt très important : il **détecte au fur et à mesure**, donc sans coût de calcul supplémentaire, **les directions importantes**. Dans une direction donnée, l'algorithme raffine en plaçant des points supplémentaires jusqu'à ce qu'il estime que la fonction est connue dans cette direction, et il continue à augmenter les niveaux dans les directions difficiles à approcher.

Les directions où l'algorithme raffine le plus correspondent aux directions qui présentent des variations plus importantes que pour les autres : le critère du niveau obtenu à la fin dans chaque direction sert à effectuer un classement des paramètres par ordre d'importance :

- niveau 1 : le paramètre est considéré négligeable,
- puis on compare les niveaux supérieurs à 1 entre eux pour éventuellement classer les paramètres en principaux et secondaires.

Les niveaux étant des nombres entiers petits, le classement est nécessairement moins fin que quand on compare la distribution des gradients. En particulier, les paramètres auront souvent le même niveau à l'intérieur d'une même catégorie.

Le résultat de cette méthode est une sortie texte que nous présenterons sous forme de tableaux (voir par exemple tableau VI.12 page 273).

Paramètre	Nom	Caractéristique	Intervalle de définition	Loi de probabilité
p_1	h-cnp	hauteur de canopée	[0 ;10]	Uniforme
p_2	cloud-cover	couverture nuageuse	[0 ;1]	Uniforme
p_3	wind-speed	vitesse du vent	[1 ;10]	Uniforme
p_4	wind-dir	direction du vent	[240 ;300]	Uniforme
p_5	z-rel	hauteur de rejet	[0 ;1]	Uniforme
p_6	c-mass	masse de produit	[10 ;300]	Uniforme
p_7	w-mom	vitesse de rejet	[200 ;800]	N(500,100)
p_8	buoy	température de rejet	[400 ;1000]	N(700,100)
p_9	sltrop	échelle de turbulence	[10 ;50]	N(30,7)

TAB. VI.3 – Cas test à 9 paramètres : scénario

3 Cas test à 9 paramètres

3.1 Hypothèses

Nous présentons dans le tableau VI.3 le scénario associé au cas test où 9 paramètres sont considérés incertains, défini par

- le nom et la nature des paramètres dans le logiciel SCIPUFF,
- leur intervalle de définition,
- la loi de probabilité associée à leur incertitude.

3.2 Réseaux de neurones

3.2.1 Mise en oeuvre

Nous précisons ici les paramètres de construction du réseau neuronal.

Nous avons vu que dans le cas de l'algorithme utilisé, (voir page 219), les deux paramètres de construction du réseau neuronal qui ont une influence sur l'approximation sont :

- la tolérance τ (qui est un coefficient utilisé dans l'algorithme pour déterminer le critère d'arrêt pour la norme des résidus entre G et g),
- et le nombre de données d'apprentissage.

Comme nous voulons réduire le plus possible le coût lié aux appels à G, nous avons fixé plusieurs tailles de lots d'apprentissage, et pour chacune nous avons testé plusieurs valeurs de tolérance pour obtenir le meilleur réseau neuronal, en comparant les erreurs sur le même lot de vérification, obtenu par un tirage de type carré latin de 1000 points.

	Lot d'apprentissage	Neurones cachés	$CorA$	$CorV$	$EQMA$	$EQMV$
g_{R-500}	500	12	0.96	0.88	0.035	0.120

TAB. VI.4 – Réseau neuronal g_{R-500} : erreurs de validation (lot de vérification : 1000 points)

3.2.2 Réseau de neurones obtenu

Nous avons obtenu une bonne approximation avec un tirage de 500 points d'apprentissage, avec $\tau = 10^{-2}$, le graphique de validation du réseau neuronal noté g_{R-500} est présenté dans la figure VI.7.

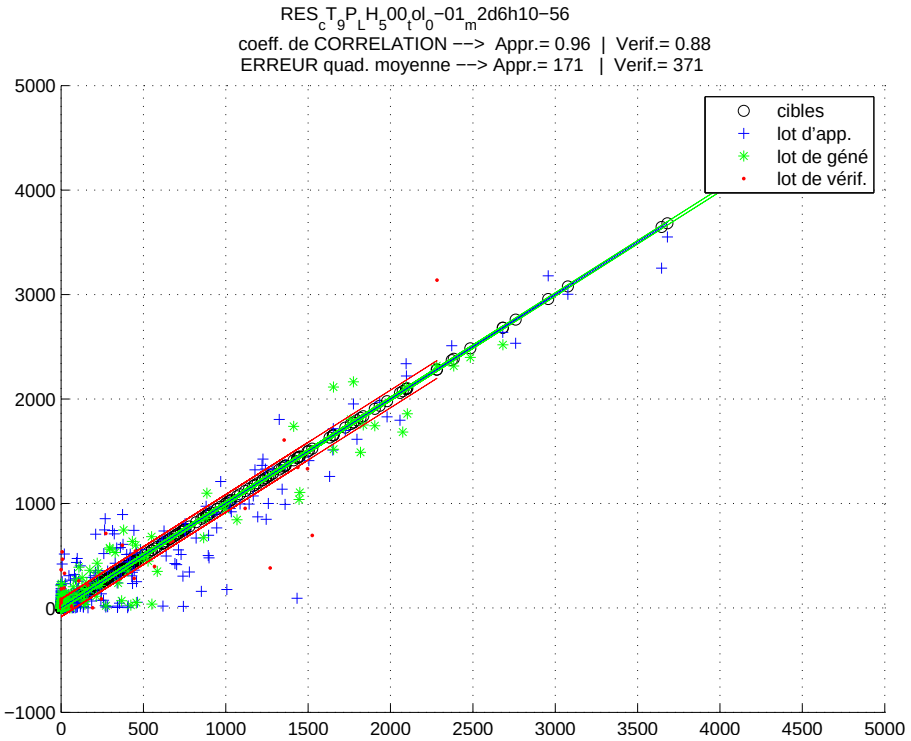


FIG. VI.7 – Réseau neuronal g_{R-500} : graphique de validation

Le nombre de neurones cachés et les critères de validation sont présentés aussi dans le tableau VI.4.

Les coefficients de corrélation montrent un bon apprentissage et une qualité de prédiction inférieure.

Mais comme $2^9 = 512$, avec un nombre de données de 500 points en dimension 9, nous avons pour la construction du réseau neuronal l'équivalent de deux points par direction, et la généralisation ne peut pas être aussi précise que l'apprentissage.

L'amplitude du nombre de victimes sur le lot d'apprentissage varie entre 0 et 4816.

3.2.3 Probabilités

Le graphique du calcul des probabilités est présenté dans la figure VI.8 : la courbe obtenue avec g_{R-500} reste presque entièrement dans l'intervalle de confiance de la courbe de référence, le résultat est très satisfaisant.

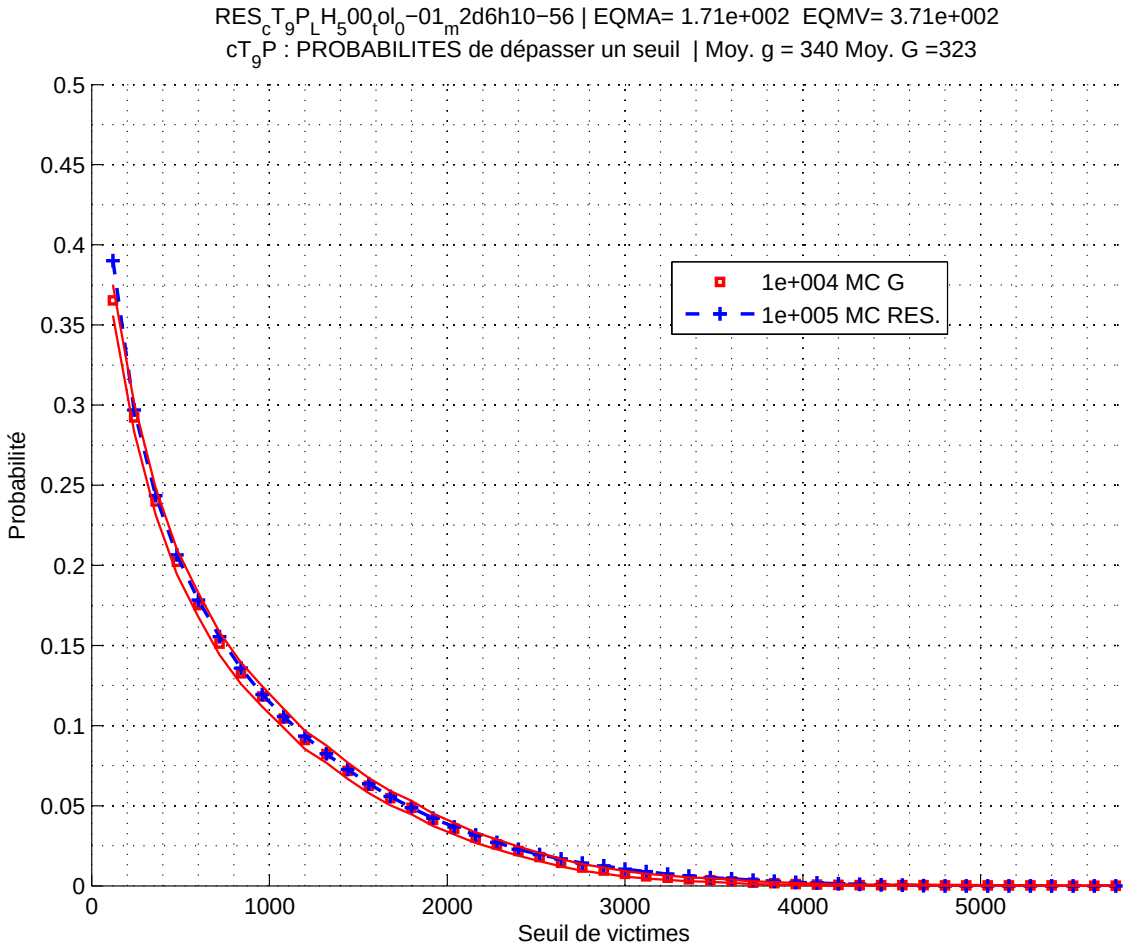


FIG. VI.8 – Réseau neuronal g_{R-500} : courbe des probabilités de dépasser un seuil

```
-----------------------------------------------------------------
  Résultats du calcul des points de CONCEPTION:
-----------------------------------------------------------------
  Seuil de victimes:  50
.................................................................
 Param. Princ.:       1   2   3   4   5   6   7   8   9
 Lois:                U   U   U   U   U   U   N   N   N
 Moyenne (LN):        0   0   0   0   0   0 500 700  30
-----------------------------------------------------------------
 Résidus fonction coût:
       1e-008   2e-006   1e-007   4e-008   1e-007
-----------------------------------------------------------------
  COORDONNEES des points de CONCEPTION P*:
.................................................................
         P*1      P*2      P*3      P*4      P*5
p_1      3.38     2.43     4.96     9.20     7.55
p_2      0.13     0.53     1.00     0.18     0.37
p_3      3.51     3.66     7.54     3.24     9.72
p_4    256.80   298.89   279.96   247.92   262.37
p_5      0.74     0.47     0.10     0.39     0.23
p_6     86.70   247.59   107.50   186.75   232.76
p_7    500.01   500.05   499.98   500.01   499.97
p_8    699.99   700.12   700.02   700.02   699.98
p_9     30.00    30.00    30.00    30.00    30.00
.................................................................
g(P*)=  50       50       50       50       50
G(P*)=  59       73       39       60      208
-----------------------------------------------------------------
 Succès optim.:    50%
```

TAB. VI.5 – Réseau neuronal g_{R-500} : points de conception pour s=50

3.2.4 Points de conception

Seuil $s = 50$ Nous recherchons les points de conception pour un seuil bas $s = 50$, ce qui dans le cadre de notre étude semble naturel (la réponse étant un nombre de victimes d'accident) : le résultat est présenté dans le tableau VI.5.

Les points sont bien au bord du domaine de défailllance associé à l'approximation g, car $g(P_i^*) = 50$, mais pas toujours au bord du domaine de défaillance D_s associé à G, car $G(P_i^*) \neq 50$.

Nous remarquons en ce qui concerne les coordonnées des points de conception trouvés :

- elles diffèrent pour les paramètres de loi uniforme,
- par contre elles sont égales pour les paramètres de loi gaussienne, et valent la moyenne correspondante.

Nous trouvons ainsi plusieurs points, et l'interprétation des résultats est difficile.

Seuil $s = 50$, lois gaussiennes Nous gardons le même seuil, et le même réseau neuronal, mais **pour trouver un seul point de conception**, nous changeons le

```
-----------------------------------------------------------------
  Résultats du calcul des points de CONCEPTION:
-----------------------------------------------------------------
  Seuil de victimes:  50
.................................................................
 Param. Princ.:       1   2   3   4   5   6   7   8   9
 Lois:                N   N   N   N   N   N   N   N   N
 Moyenne (LN):        5   1   6 270   1 155 500 700  30
-----------------------------------------------------------------
  Résidus fonction coût:
      8e+000   8e+000   8e+000   8e+000   8e+000
-----------------------------------------------------------------
  COORDONNEES des points de CONCEPTION P*:
.................................................................
         P*1      P*2      P*3      P*4      P*5
p_1      4.92     4.92     4.92     4.92     4.92
p_2      0.57     0.57     0.57     0.57     0.57
p_3      3.65     3.65     3.65     3.65     3.65
p_4    253.33   253.34   253.33   253.33   253.33
p_5      0.56     0.56     0.56     0.56     0.56
p_6    141.19   141.23   141.19   141.17   141.19
p_7    488.75   489.81   488.75   488.27   488.75
p_8    692.23   692.14   692.23   692.26   692.23
p_9     29.32    29.32    29.32    29.32    29.32
.................................................................
g(P*)=  50       50       50       50       50
G(P*)=  53       77       53       63       53
-----------------------------------------------------------------
 Succès optim.:   100%
```

TAB. VI.6 – Réseau neuronal g_{R-500} : points de conception pour s=50, lois gaussiennes

scénario en appliquant des **lois gaussiennes pour tous les paramètres**, comme indiqué dans les résultats présentés dans le tableau VI.6..

Les résidus de la fonction coût à la fin de l'algorithme ont augmenté (les points trouvés sont moins probables), mas cette fois, les points de conception trouvés ont des coordonnées très proches quelque soit le point de départ de l'optimisation.

Cependant, nous constatons aussi avec les valeurs $G(P_i^*)$ qu'une très faible variation de certaines coordonnées entraîne des réponses très différentes. Parmi les cinq points trouvés, $P_1^* = P_2^* = P_3^*$ est le point le plus proche du bord du véritable domaine de défaillance avec $G(P_1^*) = 53$.

Paramètres					
principaux		secondaires		négligeables	
p_4	direction du vent	p_6	masse de produit	p_9	échelle de turbulence
p_3	vitesse du vent	p_1	hauteur de canopée	p_5	hauteur de rejet
				p_2	couverture nuageuse
				p_7	vitesse de rejet
				p_8	température de rejet

TAB. VI.7 – Réseau neuronal g_{R-500} : classement des paramètres

3.2.5 Paramètres importants

Nous appliquons l'algorithme présenté page 256 : la distribution des gradients calculés avec g_{R-500} sur un tirage de type carré latin de 1000 points est affichée dans la figure VI.9.

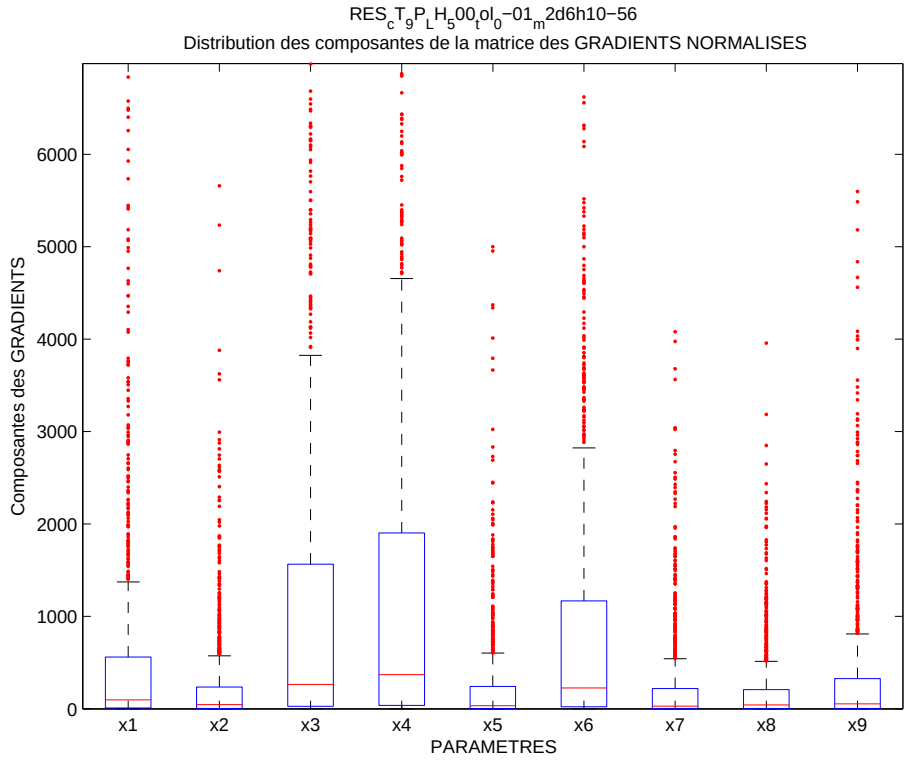


FIG. VI.9 – Réseau neuronal g_{R-500} : distribution des gradients pour classement des paramètres

D'après la distribution des gradients, nous pouvons classer les paramètres par ordre d'importance : nous avons choisi de les placer dans trois catégories (principaux, secondaires et négligeables), et par ordre d'importance à l'intérieur de chaque catégorie : le résultat est présenté dans le tableau VI.7.

Le classement par ordre d'importance est un résultat objectif de la méthode, mais la distinction entre les trois catégories est en partie subjective. Ainsi, le paramètre

p_6 de masse de produit, troisième par ordre d'importance, pourrait être placé dans la catégorie "paramètres principaux", après p_4 et p_3.

Interprétation physique Le classement est cohérent avec la physique du problème.

En effet, étant donné la distribution de la population dans deux rectangles, le nombre de victimes sera différent de zéro si le nuage toxique atteint ces rectangles.

Or la forme du nuage est déterminée essentiellement dans notre scénario, par ordre d'importance, par les variations de la direction p_4 et de la vitesse p_3 du vent (paramètres principaux).

Ensuite (paramètres secondaires), une fois que les conditions pour que le nuage arrive sur la population sont présentes, il est normal que la masse de produit p_6 influe sur le nombre de victimes.

Une hauteur de canopée p_2 importante doit retenir une partie des retombées toxiques sur le sl et limiter ainsi le nombre de victimes par rapport à une hauteur de canopée faible.

Les paramètres restants du scénario (négligeables) sont d'interprétation physique plus délicate.

Coupes 1D Nous présentons dans la figure VI.10 les coupes de G et du réseau neuronal suivant chaque direction en un point où le nombre de victimes est élevé.

Ces coupes permettent de vérifier sur un exemple le comportement de G décrit précédemment : les variations les plus importantes correspondent bien aux paramètres principaux et secondaires détectés.

On vérifie aussi la propriété qui justifiait le recours au gradient de g pour estimer les variations de G : l'approximation g lisse le comportement de G en effaçant les petites variations locales.

L'approximation g_{R-500} n'est pas toujours précise car l'information est insuffisante (le nombre de données est faible par rapport à la dimension) pour espérer approcher correctement G sur tout le domaine, mais le réseau neuronal capture au moins les tendances globales, qui suffisent pour déterminer les paramètres importants.

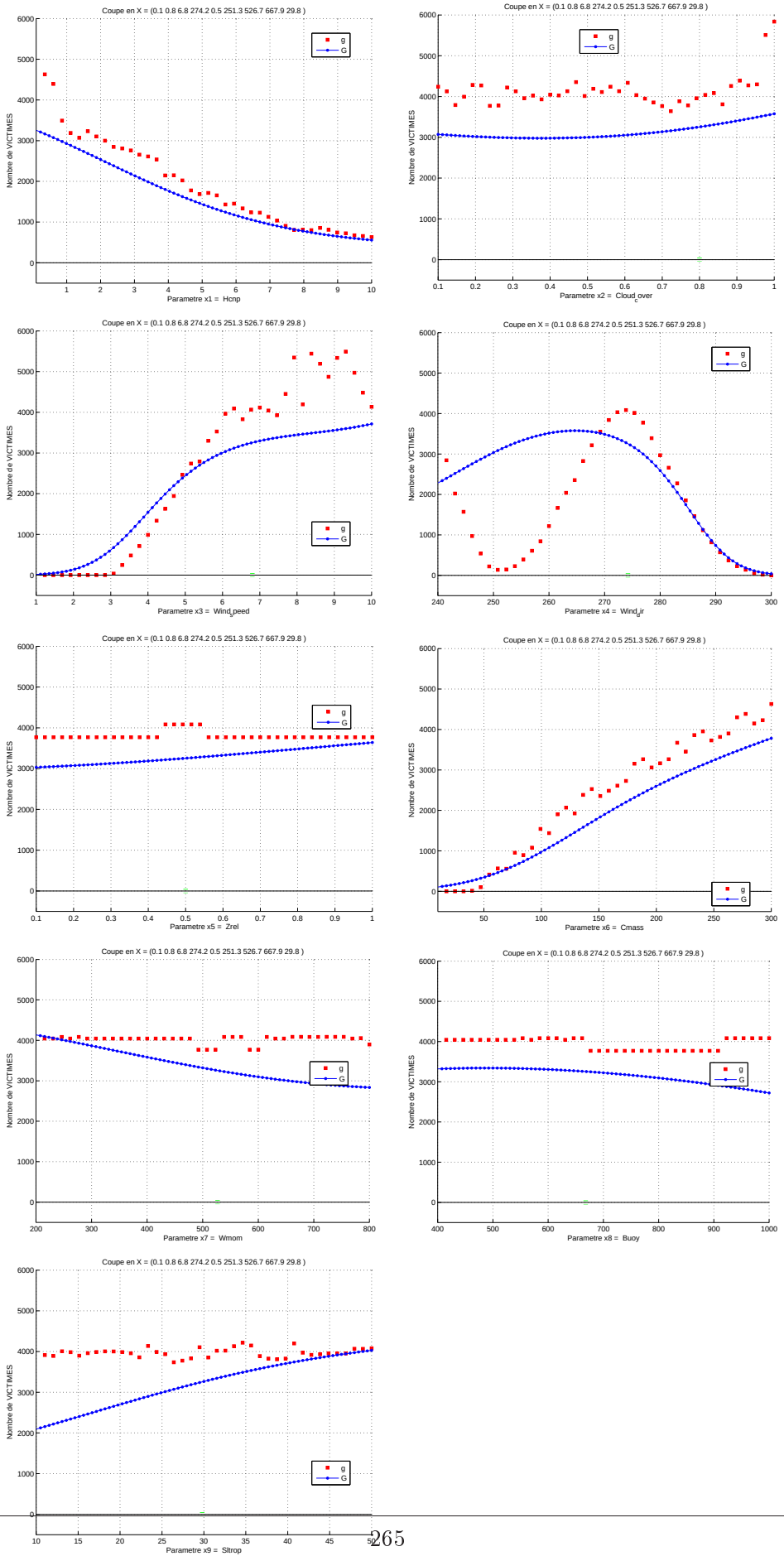


FIG. VI.10 – Réseau neuronal g_{R-500} et G : coupes 1D

Niveau	Type	Nombre de points	E_R	$EQMV$
2	Clenshaw-Curtis	181	0.5207	0.1247
	Chebyshev	181	0.5207	0.1504
3	Clenshaw-Curtis	1177	0.5687	0.1176
	Chebyshev	1177	0.5687	0.1570

TAB. VI.8 – Cas non adaptatif : erreurs sur les *sparse grid*s

3.3 *Sparse grids*

3.3.1 Mise en oeuvre

La construction d'une *sparse grid* présente l'avantage de la simplicité : nous définissons seulement le nombre de points de grille désiré.

L'erreur relative estimée par l'algorithme est notée E_R, et l'erreur quadratique $EQMV$ est calculée sur un tirage carré latin de vérification de 1000 points (voir section 2.4.2 page 252).

3.3.2 Mode non adaptatif

Nous présentons d'abord le cas non adaptatif : nous fixons le niveau de grille qui détermine le nombre de points, de façon à obtenir un nombre d'évaluations de G proche de la taille des données, égale à 500 dans le cas du réseau de neurones g_{R-500} obtenu pour le même cas test.

Nous présentons dans le tableau VI.8 les différentes *sparse grids* et les erreurs associées obtenues : nous avons utilisé les niveaux 2 et 3, qui donnent un nombre de points respectivement inférieur et supérieur à 500.

Du point de vue de la courbe des probabilités, le meilleur résultat est obtenu avec la *sparse grid* qui présente l'erreur $EQMV$ la plus petite, c'est-à-dire la *sparse grid* de type Clenshaw-Curtis de niveau 3 à 1177 points, notée g_{S_1} (voir figure VI.11).

Nous remarquons ici que l'erreur $EQMV$ sur un lot de vérification semble être un critère de précision meilleur que l'erreur E_R détectée par les surplus, puisque cette dernière augmente quand l'erreur de vérification diminue.

3.3.3 Mode adaptatif

***Sparse grids* obtenues** Nous allons utiliser le cas adaptatif de l'algorithme qui devrait nous donner un résultat similaire au précédent pour un coût moindre, avec l'avantage de nous présenter aussi les directions importantes.

Ici nous fixons le nombre maximum de points de grille, noté N_{max}. Si nous fixons une tolérance très basse qui ne sera pas atteinte, l'algorithme choisit adaptativement des niveaux différents suivant les directions (élevés pour les directions importantes) jusqu'à atteindre ce nombre maximum (le nombre de points de grille finalement obtenu diffère légèrement de l'ordre de 10 à 50 points par rapport à N_{max}).

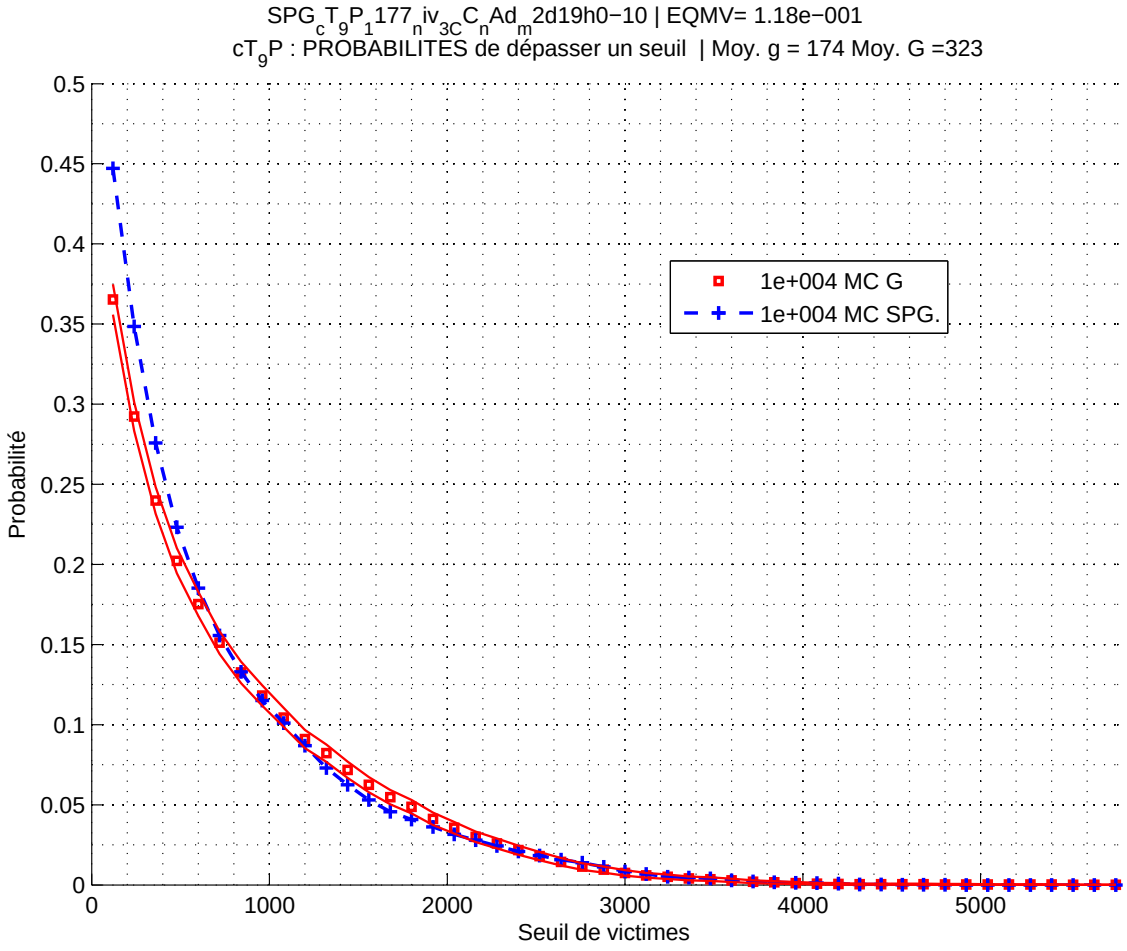


FIG. VI.11 – *Sparse grid* g_{S_1} : courbe des probabilités de dépasser un seuil

Nous présentons les *sparse grids* et les erreurs obtenues en choisissant $N_{max} = 100, 300, 500$ dans le tableau VI.9 avec le même tirage de vérification que dans le cas non adaptatif.

Nous constatons que le cas adaptatif permet d'améliorer l'erreur de vérification pour un coût inférieur.

Probabilités Nous remarquons ici que la précision de la courbe de probabilités n'est pas directement liée à l'erreur de la *sparse grid*.

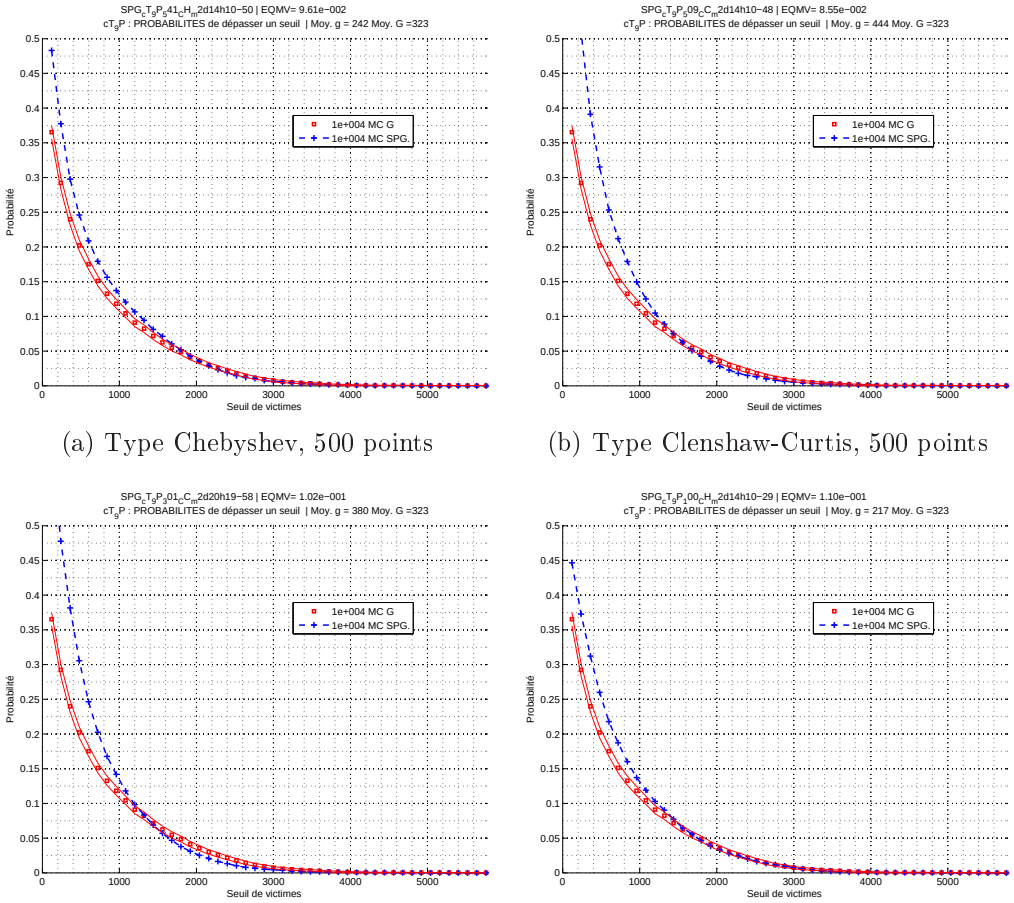
En effet, sur la figure VI.12(a), la courbe de la *sparse grid* de type Chebyshev avec 500 points est plus proche de la courbe de référence que celle de Clenshaw-Curtis avec 500 points sur VI.12(b) et avec 300 points sur VI.12(c), alors que les erreurs sont similaires.

D'autre part, sur la figure VI.12(d), la courbe de probabilités de la *sparse grid* de type Chebyshev est de même précision pour 100 points que pour 500.

Conclusion Le cas adaptatif permet donc d'obtenir le même résultat avec 100 points en ce qui concerne les probabilités que le cas non adaptatif avec 1177 points,

N_{max}	Type	E_R	$EQMV$
100	Clenshaw-Curtis	0.2313	0.1209
	Chebyshev	0.2678	0.1101
300	Clenshaw-Curtis	0.0901	0.1020
	Chebyshev	0.2964	0.1324
500	Clenshaw-Curtis	0.1239	0.0855
	Chebyshev	0.3106	0.0961

TAB. VI.9 – *Sparse grids* - cas adaptatif : erreurs de validation



(a) Type Chebyshev, 500 points

(b) Type Clenshaw-Curtis, 500 points

(c) Type Clenshaw-Curtis, 300 points

(d) Type Chebyshev, 100 points

FIG. VI.12 – *Sparse grids* - cas adaptatif : courbe des probabilités de dépasser un seuil

ce qui représente un gain considérable.

Les meilleurs résultats sont obtenus avec les fonctions de base polynomiales (type Chebyshev) que les fonctions de base linéaires par morceaux (type Clenshaw-Curtis) : elles semblent mieux adaptées à l'approximation d'une fonction non régulière comme celle du cas test.

Dans la suite, nous utilisons les *sparse grids* précédentes les plus parcimonieuses qui ont donné des résultats corrects pour le calcul de probabilités pour essayer de

retrouver les résultats obtenus avec le réseau neuronal g_{R-500}.

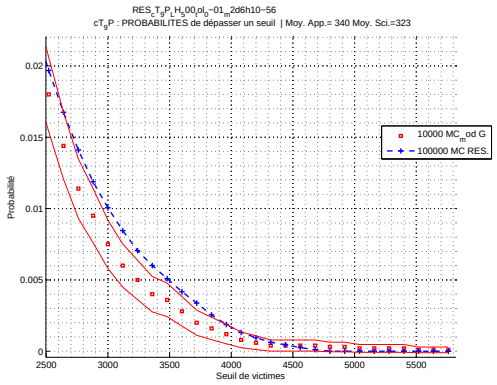
Nous notons donc $g_{S_{CC-100}}$ et $g_{S_{CH-100}}$ les *sparse grids* adaptatives à 100 points respectivement de type Clenshaw-Curtis et Chebyshev.

Remarque : probabilités faibles Dans les courbes précédentes, les probabilités calculées avec le réseau neuronal g_{R-500} sont meilleures que celles calculées avec les *sparse grids* $g_{S_{CC-100}}$ et $g_{S_{CH-100}}$ sur les seuils bas (probabilités importantes).

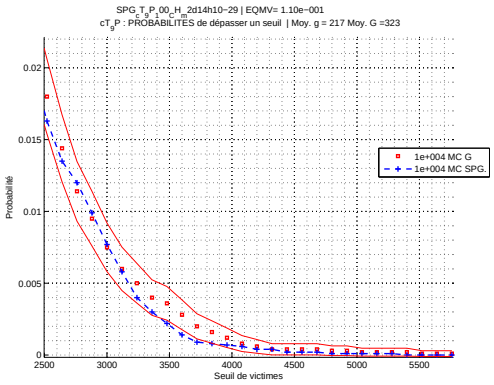
Par contre, commme le montrent les courbes de la figure VI.13 les *sparse grids* donnent de meilleurs résultats pour les seuils hauts (probabilités faibles).

La disposition dans le domaine de définition des données d'apprentissage est en effet très différente dans les deux méthodes, ce qui peut expliquer des performances différentes des approximations :

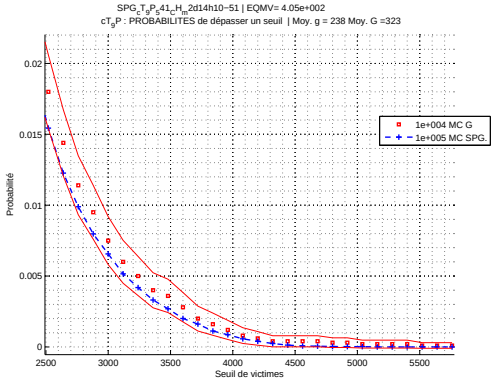
- Pour le réseau neuronal, les données sont issues d'un tirage de type hypercube latin qui concentre l'information autour du point nominal (point d'intersection des moyennes des lois gaussiennes). Dans ce cas test, le point nominal est au centre du domaine.
- Pour les *sparse grids*, les donnnées sont concentrées sur quelques axes passant par le centre mais aussi vers les bords du domaine.



(a) Réseau neuronal g_{R-500}



(b) *Sparse grid* $g_{S_{CC-100}}$



(c) *Sparse grid* $g_{S_{CH-100}}$

FIG. VI.13 – Comparaison des probabilités faibles pour le réseau neuronal et les *sparse grids*

```
-----------------------------------------------------------------
  Résultats du calcul des points de CONCEPTION:
-----------------------------------------------------------------
  Seuil de victimes:  50
.................................................................
 Param. Princ.:       1   2   3   4   5   6   7   8   9
 Lois:                N   N   N   N   N   N   N   N   N
 Moyenne (LN):        5   1   6 270   1 155 500 700  30
-----------------------------------------------------------------
 Résidus fonction coût:
      2e+000   1e+000   1e+000   2e+000   1e+000
-----------------------------------------------------------------
  COORDONNEES des points de CONCEPTION P*:
.................................................................
         P*1      P*2      P*3      P*4      P*5
p_1      5.28     5.27     5.27     5.28     5.27
p_2      0.55     0.55     0.55     0.55     0.55
p_3      6.52     6.31     6.30     6.52     6.31
p_4    274.10   264.80   264.80   274.10   264.80
p_5      0.55     0.55     0.55     0.55     0.55
p_6    126.81   132.49   132.48   126.81   132.49
p_7    500.00   500.02   502.19   502.76   500.01
p_8    699.75   699.95   698.09   700.00   699.98
p_9     29.91    29.93    29.94    29.91    29.96
.................................................................
g(P*)=  50       50       50       50       50
G(P*)= 134      152      160      132      151
-----------------------------------------------------------------
 Succès optim.:    15%
```

TAB. VI.10 – *Sparse grid* $g_{S_{CC-100}}$: points de conception pour s=50, lois gaussiennes

Points de conception

Seuil $s = 50$ Nous reprenons le seuil $s = 50$ utilisé avec le réseau de neurones, et nous appliquons le même algorithme de recherche de points de conception pour des lois gaussiennes sur tous les paramètres aux *sparse grids* $g_{S_{CC-100}}$ et $g_{S_{CH-100}}$.

Dans le cas de la *sparse grid* $g_{S_{CC-100}}$, présenté dans le tableau VI.10, l'algorithme trouve des points tous différents dont les coordonnées sont voisines, mais le taux de succès des optimisations est faible (15%).

Dans le cas de la *sparse grid* $g_{S_{CH-100}}$, présenté dans le tableau VI.11, deux points différents mais voisins ont été trouvés, avec un taux de succès des optimisations plus satisfaisant (71%). Nous pouvons retenir comme point de conception celui qui correspond au résidu le plus petit (c'est aussi celui qui est trouvé le plus souvent par l'algorithme).

Etant donné la différence entre les taux de succès et le nombre de points de conception trouvés pour $g_{S_{CC-100}}$ et $g_{S_{CH-100}}$, nous constatons de nouveau que les fonctions de base de type Chebyshev sont mieux adaptées dans le cadre de notre étude.

Conclusion Dans tous les cas les résidus sont inférieurs à ceux associés aux points de conception trouvés avec le réseau neuronal g_{R-500} (voir section 3.2.4 page 261),

```
-----------------------------------------------------------------
  Résultats du calcul des points de CONCEPTION:
-----------------------------------------------------------------
  Seuil de victimes:  50
.................................................................
 Param. Princ.:       1   2   3   4   5   6   7   8   9
 Lois:                N   N   N   N   N   N   N   N   N
 Moyenne (LN):        5   1   6 270   1 155 500 700  30
-----------------------------------------------------------------
 Résidus fonction coût:
      1e+000   3e+000   1e+000   1e+000   1e+000
-----------------------------------------------------------------
  COORDONNEES des points de CONCEPTION P*:
.................................................................
         P*1      P*2      P*3      P*4      P*5
p_1      5.57     5.97     5.57     5.57     5.57
p_2      0.54     0.53     0.54     0.54     0.54
p_3      5.53     5.71     5.53     5.53     5.53
p_4    262.75   280.18   262.75   262.75   262.75
p_5      0.55     0.55     0.55     0.55     0.55
p_6    133.65   120.40   133.66   133.65   133.66
p_7    502.42   504.97   502.62   502.50   502.84
p_8    697.94   696.72   697.97   698.05   698.49
p_9     29.73    29.51    29.73    29.73    29.73
.................................................................
g(P*)=  50       50       50       50       50
G(P*)= 140      129      140      140      140
-----------------------------------------------------------------
 Succès optim.:    71%
```

TAB. VI.11 – *Sparse grid* $g_{S_{CH-100}}$: points de conception pour s=50, lois gaussiennes

Paramètres		p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9
Niveaux	$g_{S_{CH-100}}$	2	1	2	3	1	2	1	1	1
	$g_{S_{CH-500}}$	3	2	4	3	1	2	1	1	1

TAB. VI.12 – *Sparse grids* : niveaux dans chaque direction obtenus par le mode adaptatif

ce qui indique que les points de conception trouvés avec la *sparse grid* $g_{S_{CH-100}}$ sont plus probabless.

Mais d'autre part, l'image par G des points de conception les plus probables (tous sauf P_2^*), égale à 140, est beaucoup plus éloignée du seuil de 50 que pour le réseau neuronal : les résultats sont moins satisfaisants, et ne sont pas améliorés si on utilise des *sparse grids* à 500 points, ils sont même dégradés.

Paramètres importants

Niveaux du mode adaptatif Les niveaux obtenus sont présentés dans le tableau VI.12 : avec la *sparse grid* $g_{S_{CH-100}}$, nous avons aussi affiché les niveaux obtenus avec la *sparse grid* de type Chebyshev à 500 points notée $g_{S_{CH-500}}$.

Le classement que l'on obtient est présenté dans le tableau VI.13 pour $g_{S_{CH-100}}$ et dans le tableau VI.14 pour $g_{S_{CH-500}}$.

Les résultats sont similaires à ceux du tableau VI.7 pour la méthode des gradients avec le réseau neuronal : la *sparse grid* $g_{S_{CH-100}}$ permet d'obtenir presque le même classement en trois catégories, mais sans pouvoir classer plus finement les paramètres comme évoqué dans la section 2.7.4 (voir page 257).

Nous pouvons souligner les différences suivantes :

- la vitesse du vent p_3, paramètre principal d'après les gradients du réseau neuronal, est classée comme secondaire par la *sparse grid* à 100 points, mais on retrouve le classement en principal si on passe à 500 points,
- la hauteur de canopée p_1, paramètre secondaire d'après les gradients du réseau neuronal, est classée comme principale par la *sparse grid* à 500 points,
- la couverture nuageuse p_2, paramètre négligeable d'après les gradients du réseau neuronal, est classée comme secondaire par la *sparse grid* à 500 points.

Nous constatons que si on augmente le nombre de points de la *sparse grid* (ici de 100 à 500), les niveaux augmentent forcément dans certaines directions, importantes ou secondaires, ce qui peut modifier le classement des paramètres.

Distribution des gradients Nous pouvons aussi appliquer la méthode des gradients pour un coût de calcul négligeable avec les *sparse grids* calculées : cela permet d'obtenir une information plus riche que les seuls niveaux de grille, et de valider le classement des paramètres en croisant les résultats.

Paramètres					
principaux		secondaires		négligeables	
p_4	direction du vent	p_3	vitesse du vent	p_9	échelle de turbulence
		p_6	masse de produit	p_5	hauteur de rejet
		p_1	hauteur de canopée	p_2	couverture nuageuse
				p_7	vitesse de rejet
				p_8	température de rejet

TAB. VI.13 – *Sparse grid* $g_{S_{CH-100}}$: classement des paramètres

Paramètres					
principaux		secondaires		négligeables	
p_3	vitesse du vent	p_6	masse de produit	p_9	échelle de turbulence
p_4	direction du vent	p_2	couverture nuageuse	p_5	hauteur de rejet
p_1	hauteur de canopée			p_7	vitesse de rejet
				p_8	température de rejet

TAB. VI.14 – *Sparse grid* $g_{S_{CH-500}}$: classement des paramètres

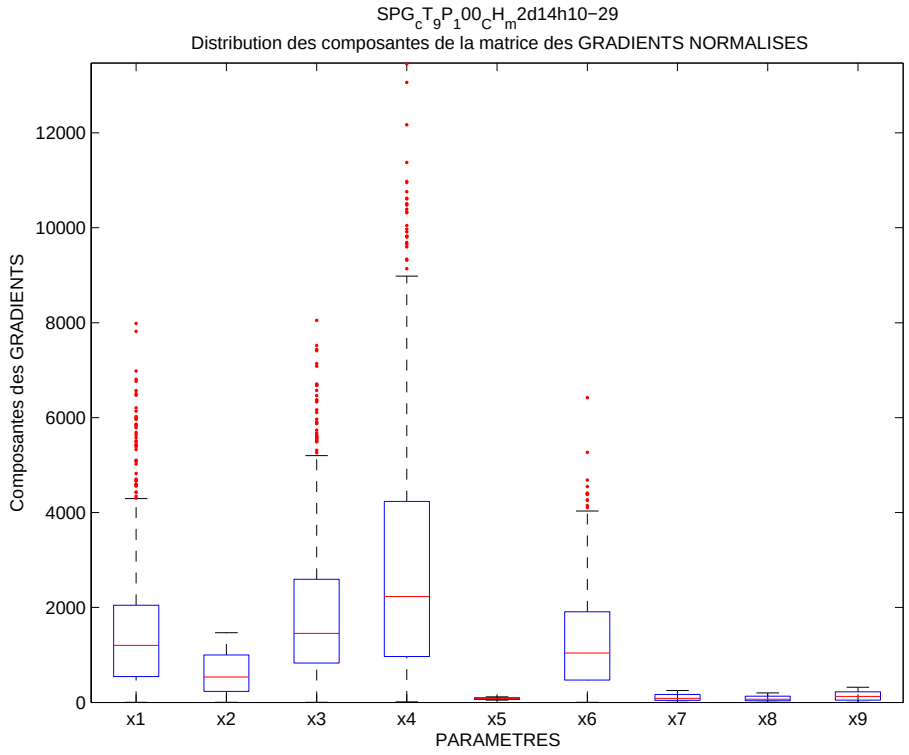


FIG. VI.14 – *Sparse grid* $g_{S_{CH-100}}$: distribution des gradients pour classement des paramètres

La figure VI.14 présente la distribution des gradients pour la *sparse grid* $g_{S_{CH-100}}$.

Nous retrouvons exactement le même classement que celui donné par les niveaux

$g_{S_{CH-100}}$ dans le tableau VI.13, mais avec l'avantage de pouvoir ici classer tous les paramètres entre eux.

3.4 Conclusion

Dans le cadre de ce cas test en dimension $n = 9$, nous avons utilisé une approximation globale du modèle initial :

- L'approximation par **réseau de neurones** donne des résultats corrects pour les trois problèmes (P_i) à résoudre, pour un coût global de 500 évaluations de la fonction de modélisation par rapport à la dimension (pour une grille pleine de même taille, cela correspond à deux points dans chaque direction).

  Cette technique présente l'avantage de pouvoir placer les données de construction avec un plan d'expériences dans les zones d'intérêt, mais la phase d'apprentissage peut s'avérer délicate, et nous obtenons des approximations légèrement différentes à chaque fois.

- L'utilisation des ***sparse grids*** en mode adaptatif donne aussi des résultats satisfaisants pour un **coût de calcul réduit**.

  Avec 100 évaluations au total seulement, les résultats obtenus avec la *sparse grid* $g_{S_{CH-100}}$ sont similaires à ceux obtenus avec le réseau neuronal g_{R-500} en ce qui concerne les probabilités et les directions importantes.

  En ce qui concerne les points de conception, l'algorithme d'optimisation semble cependant rencontrer des minima locaux qui l'empêchent de converger vers le point de conception correct : il faudrait appliquer des techniques de **régularisation** aux *sparse grids* utilisées afin de passer d'une interpolation des données à approximation plus "lisse" et **améliorer les résultats**.

Dans l'objectif de traiter des **problèmes en grande dimension**, nous choisissons d'appliquer la technique des ***sparse grids*** : grâce à leur **mode adaptatif**, elles permettent d'acquérir l'information nécessaire à la résolution de nos problèmes à coût très réduit, en **concentrant les évaluations dans les directions difficiles à approcher**.

Ainsi pour le cas test suivant en dimension $n = 30$, nous laissons de côté l'approximation par réseau de neurones pour nous concentrer sur les perspectives encourageantes des *sparse grids* en ce qui concerne le coût de calcul.

Comme cette technique est émergente, elle nécessite peut être des améliorations pour donner des résultats corrects dans le cadre de notre étude (comme dans le cas des points de conception).

Paramètre	Nom	Intervalle de définition	Loi de probabilité
p_1	Zrel	[0.5 ;10]	Uniforme
p_2	Cmass	[10 ;100]	Uniforme
p_3	Size	[5 ;10]	Uniforme
p_4	Tdur	[5.10^{-4} ;0.001]	Uniforme
p_5	Wmom	[5 ;50]	Uniforme
p_6	Buoy	[20 ;500]	Uniforme
p_7	Uu-ensm	[0 ;50]	Uniforme
p_8	Zimin	[30 ;70]	N(50,5)
p_9	Zimax	[500 ;1500]	N(1000,150)
p_{10}	Hconst	[0 ;10]	N(5,1)
p_{11}	Hdiur	[30 ;70]	N(50,5)
p_{12}	Hcnp	[0 ;15]	N(7,2)
p_{13}	Albedo	[0 ;1]	N(0.16,0.05)
p_{14}	Bowen	[0 ;1]	N(0.6,0.1)
p_{15}	Cloud-cover	[0 ;1]	Uniforme
p_{16}	Alpha-max	[0.5 ;2]	N(1,0.15)
p_{17}	Alpha-min	[10^{-4} ;0.01]	N(0.001,3.10^{-4})
p_{18}	Ac-eps	[0.001 ;0.1]	N(0.01,0.003)
p_{19}	P-eps	[10^{-6} ;10^{-4}]	N(10^{-5},3.10^{-6})
p_{20}	Tbin-met	[600 ;36000]	Uniforme
p_{21}	Cmin	[0 ;10]	N(2,0.6)
p_{22}	Delmin	[0 ;5]	Uniforme
p_{23}	Wwtrop	[0.001 ;0.1]	N(0.01,0.003)
p_{24}	Epstrop	[10^{-5} ;0.001]	N(4.10^{-4},10^{-4})
p_{25}	Sltrop	[5 ;20]	N(10,1.5)
p_{26}	Uu-calm	[0.2 ;0.3]	N(0.25,0.01)
p_{27}	Sl-calm	[500 ;1500]	N(1000,150)
p_{28}	Grdmin	[0 ;2]	N(0.25,0.07)
p_{29}	Z-dosage	[0.8 ;2]	N(1.5,0.15)
p_{30}	Density	[1 ;7]	N(3,0.6)

TAB. VI.15 – Cas test à 30 paramètres : scénario

4 Cas test à 30 paramètres : *sparse grids*

4.1 Hypothèses

Nous présentons dans le tableau VI.15 le scénario associé au deuxième cas test où 30 paramètres sont considérés incertains, défini par

- le nom des paramètres dans le logiciel SCIPUFF,
- leur intervalle de définition,
- la loi de probabilité associée à leur incertitude.

La dimension du problème a augmenté : avec le bon comportement des *sparse grids*, et surtout leur avantage au niveau du coût de calcul dans le cas test précédent, nous

avons retenu uniquement cette méthode pour la suite.

4.2 Approche globale

Nous utilisons les *sparse grids* en mode adaptatif en définissant le nombre de points de grille désiré pour construire une approximation globale g de la fonction de modélisation G pour l'utiliser à sa place pour l'étude de fiabilité.

Les résultats suivants ont été obtenus avec un nombre maximum de 3000 points de grille comme paramètre d'entrée de l'algorithme, nous appellons g_{3000} l'approximation obtenue.

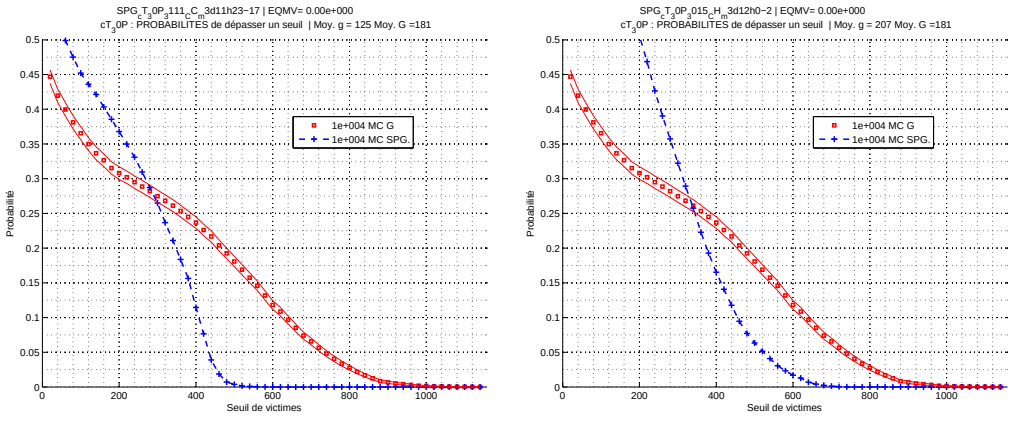
4.2.1 Résultats numériques

Probabilités La figure VI.15 nous montre :

- l'approximation globale ne donne pas une courbe de probabilités satisfaisante,
- les fonctions de base de type Chebyshev montrent de meilleurs résultats.

Le fait d'augmenter le nombre de points de grille n'améliore pas la courbe de probabilités, et à partir d'un certain nombre de points, les résultats sont même dégradés : nous retrouvons ici les phénomènes d'instabilités rencontrés dans le cadre de l'interpolation, le processus d'approximation ne converge pas avec la fonction de modélisation de notre étude.

Dans la suite nous utiliserons des fonctions de base de type Chebyshev.



(a) Type Clenshaw-Curtis (b) Type Chebyshev

FIG. VI.15 – *Sparse grid* g_{3000} : courbes des probabilités de dépasser un seuil

Points de conception En ce qui concerne les points de conception, la résolution du problème d'optimisation en dimension 30 avec des lois gaussiennes pour tous les paramètres s'avère impraticable : aucun point de conception n'est trouvé.

Paramètres importants Les niveaux maximums dans chaque direction du mode adaptatif donnent les mêmes paramètres importants que la méthode des gradients dont les résultats sont présentés dans la figure VI.16.

Seulement trois paramètres importants sur 30 sont détectés : les paramètres n°13, n°14 et n°21. Les gradients sont nuls dans les autres directions, ce qui montre que l'algorithme a détecté quelques directions suivant lesquelles la fonction G est difficile à interpoler, et a épuisé le nombre de points de grille dans ces directions pour essayer de diminuer la précision, en laissant les autres directions avec une approximation constante.

Les très grandes valeurs de gradient obtenues suivant les trois paramètres importants montrent également que l'approximation obtenue présente de fortes oscillations.

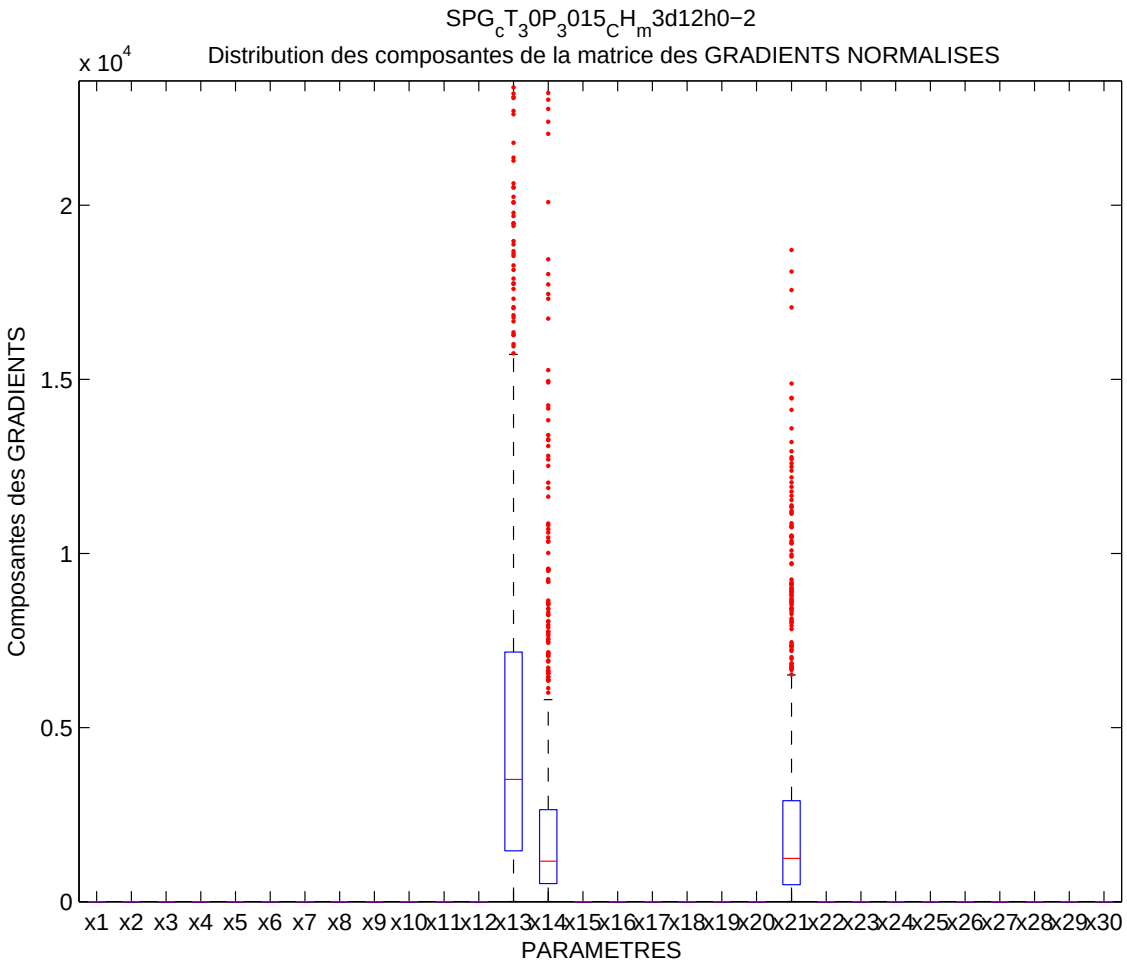
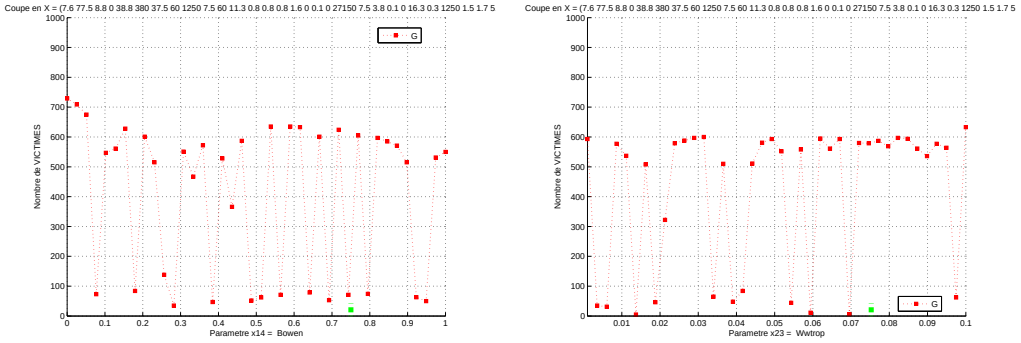


FIG. VI.16 – *Sparse grid* g_{3000} : distribution des gradients pour classement des paramètres

Irrégularité de la fonction de modélisation En observant les coupes en une dimension de la fonction de modélisation G, on remarque de fortes irrégularités (voir figure VI.17).

La chaîne de logiciels semble comporter un défaut qui provoque des réponses in-

stables : la fonction qui en résulte est difficile à approcher, d'autant plus si on utilise une méthode d'interpolation comme les *sparse grids*, ce qui explique les résultats précédents.



(a) Paramètre n°14 : Bowen (b) Paramètre n°23 : Wwtrop

FIG. VI.17 – Coupes en une dimension de la fonction G

4.2.2 Conclusion

Ce nouveau cas test nous montre les limites de l'approche globale quand le nombre de dimensions du problème augmente.

De plus, la fonction de modélisation à approcher se montre particulièrement irrégulière, ce qui rend le problème très difficile.

Nous allons présenter des changements destinés à améliorer les résultats de l'approche globale.

4.3 Améliorations

4.3.1 Probabilités : méthode avec déplacement autour du point nominal

Idée clé : déplacement de la grille La figure VI.18(a) présente une illlustration de la zone contribuant de manière prépondérante au calcul de la probabilité p_s dans l'espace gaussien standard (voir aussi section 3.4.1 page 165) : l'intersection du domaine de défaillance D_s et du support à 99% de probabilité de réalisation (hachuré), défini par les points de distance inférieure à 3 σ, avec $\sigma = 1$ l'écart-type de la loi gaussienne centrée réduite.

Si l'on note $[c_1, d_1]$ et $[c_2, d_2]$ les intervalles de variation des paramètres u_1 et u_2 dans cet espace, la figure indique comment les points de grille seront placés : cette *sparse grid* présente des points essentiellement sur les deux axes perpendiculaires qui se croisent au centre du pavé $[c_1, d_1] \times [c_2, d_2]$.

L'information dans la zone contribuant au calcul de probabilité est alors pauvre relativement à la zone proche des deux axes, notamment à leur intersection.

L'idée est donc, dans le but de calculer les probabilités, de **déplacer la grille pour avoir pour centre le point nominal**, et de **réduire les intervalles de définition** de la grille à $[-3\ \sigma, 3\ \sigma]$ dans chaque direction : on ne dispose plus d'information au-delà de cette zone, ce qui rend le calcul des probabilités très faibles moins bon, mais les autres probabilités devraient être améliorées.

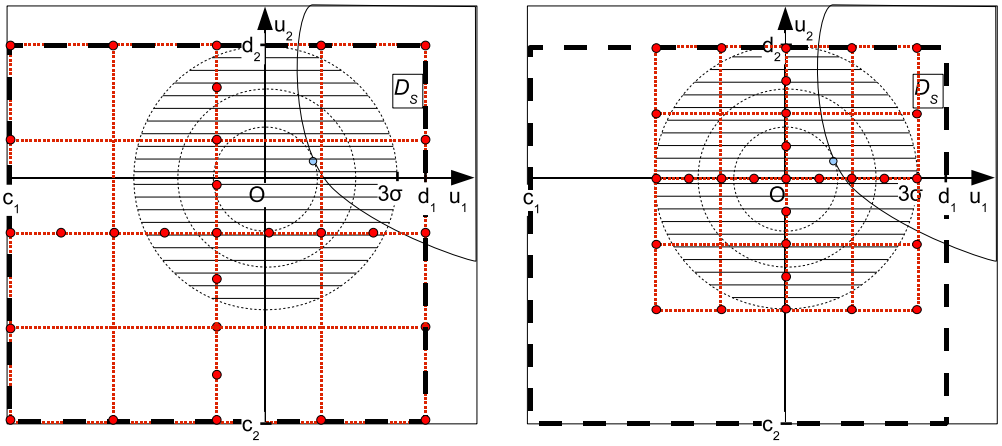
Dans le cas où l'intervalle $[-3\ \sigma, 3\ \sigma]$ déborde l'intervalle de définition $[c_i, d_i]$, nous sommes obligés de tronquer, et le centre de la grille n'est plus le point nominal.

Nous proposons d'effectuer alors un déplacement des points de grille qui ne sont pas au bord par homothétie de telle manière que les deux axes se croisent cette fois au point nominal : voir la figure VI.19 avec la grille tronquée en c_2.

Ainsi, même en grande dimension et avec peu de points de grille dans certaines directions, nous sommes assurés que le centre de la grille est le point nominal, afin d'avoir de l'information pour que la *sparse grid* soit précise autour de ce point important pour le calcul.

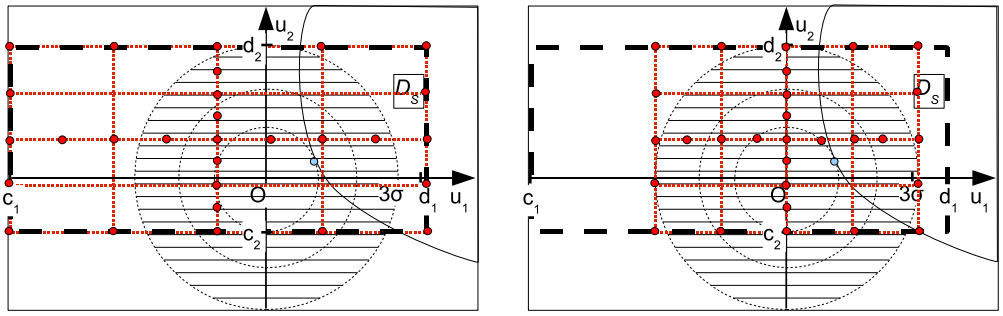
Méthode

- Prendre $[-3\ \sigma, 3\ \sigma]^n$ comme domaine de définition de la *sparse grid* autour du point nominal dans l'espace gaussien standard,
- Déplacer éventuellement à l'aide d'un changement de variables les points intérieurs de la grille, de façon à obtenir l'intersection des axes centraux sur le point nominal,
- Construire l'approximation globale *sparse grid* g de G,
- Calculer la courbe des probabilités de dépasser un seuil avec la méthode de Monte-Carlo appliquée à g.

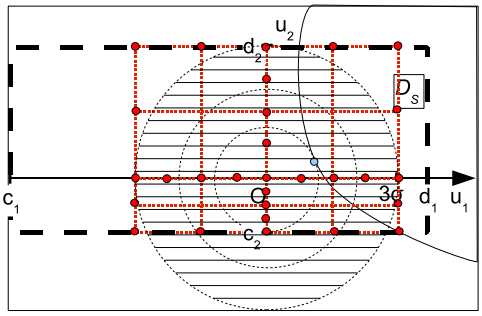


(a) Points de grille normaux (b) Points de grille déplacés

FIG. VI.18 – Déplacement des points de grille autour du point nominal



(a) Points de grille normaux (b) Points de grille déplacés



(c) Points de grille déplacés avec homothétie des points intérieurs

FIG. VI.19 – Déplacement des points de grille dans le cas tronqué

Résultats Nous construisons avec la méthode précédente une *sparse grid* en mode adaptatif à 3000 points maximum, et la courbe de probabilités obtenue de la figure VI.20 montre une nette amélioration par rapport à celle de la figure VI.15 (voir page 277) :

Cependant, le résultat n'est dans le cadre de ce cas test pas encore totalement satisfaisant, et nous présentons dans la suite une approche complémentaire.

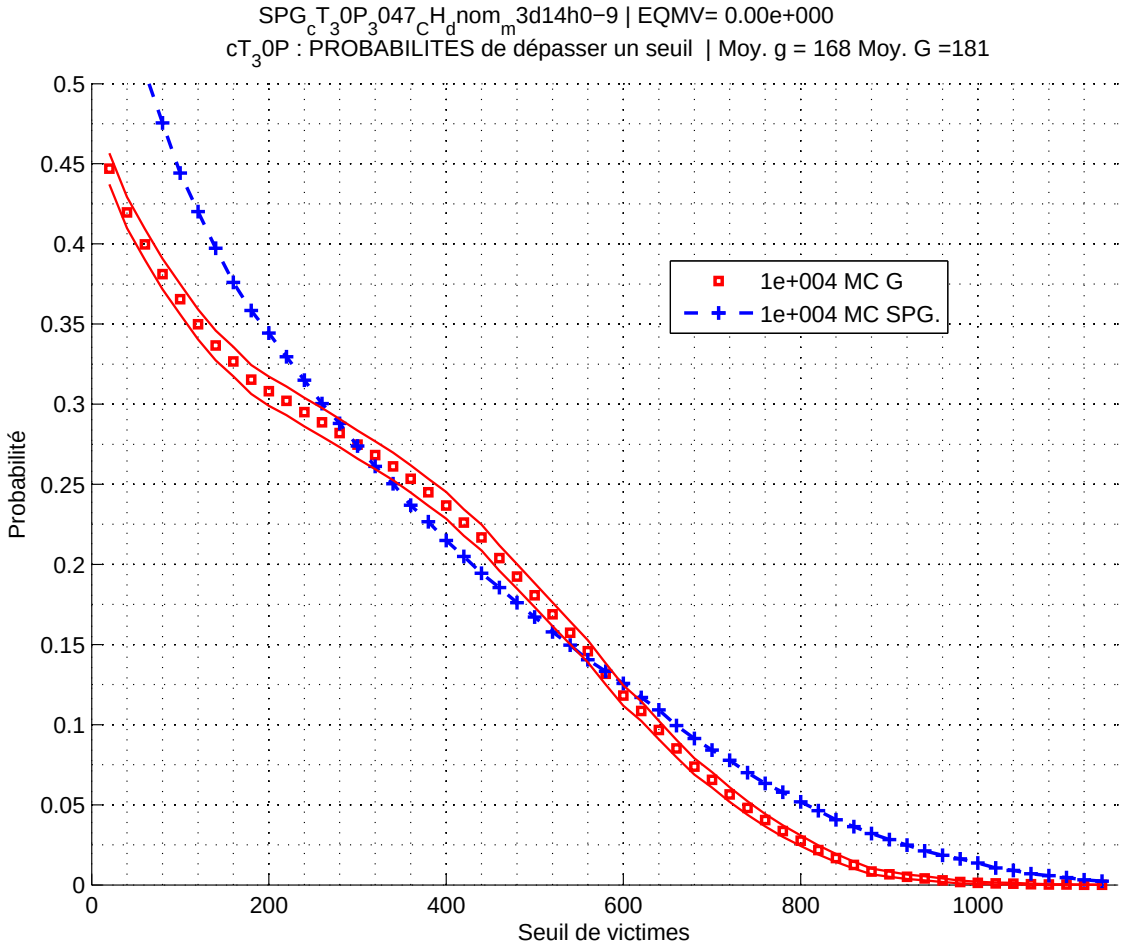


FIG. VI.20 – *Sparse grid* à 3000 points avec déplacement des points de grille vers le point nominal : courbe des probabilités de dépasser un seuil

4.3.2 Probabilités : seuillage des données

Pour avoir des résultats de probabilités plus précis, nous proposons de construire une approximation adaptée au calcul pour un seuil s donné. Cela implique que nous ne pourrons plus calculer la probabilité de dépasser n'importe quel seuil comme avec la méthode globale précédente, ce qui revient à estimer la densité de probabilité de la réponse $G(X)$, mais seulement pour le seuil choisi, comme dans les méthodes de type FORM/SORM.

Idée clé : seuillage L'idée est d'**appliquer une fonction sigmoïde Sig_s à la sortie G, de manière à séparer les valeurs de sortie entre celles qui sont inférieures au seuil s et celles qui sont supérieures** (voir aussi la section 2.1.3 page 206 du chapitre précédent).

Nous notons Sig_s la sigmoïde définie par

$$Sig_{s,\alpha}(x) = \frac{1}{1 + e^{\frac{x-s}{-\alpha}}} \tag{VI.11}$$

avec

- $s \in \mathbb{R}^+$ le seuil choisi,
- et $\alpha \in \mathbb{R}^+$ le coefficient qui détermine la pente de la sigmoïde.

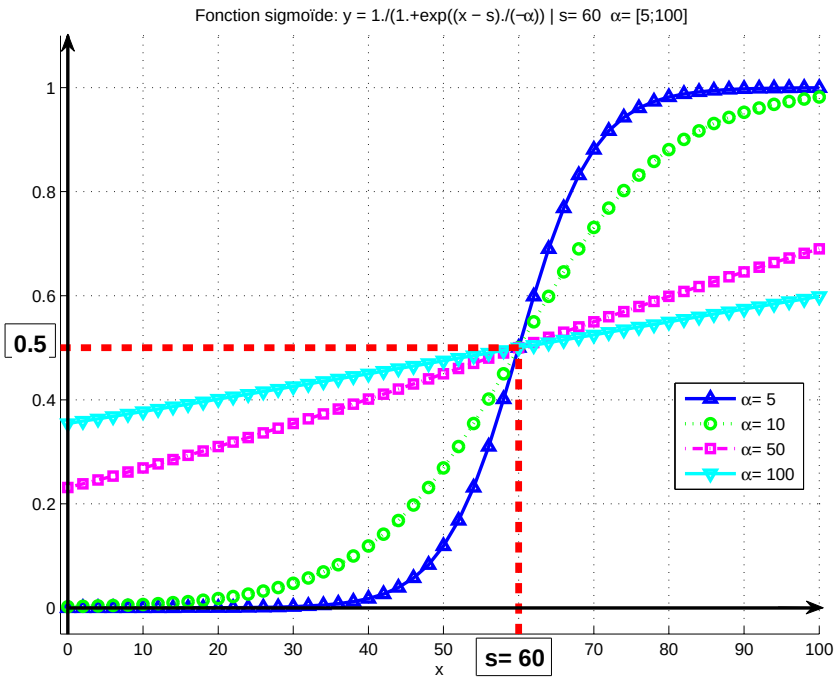


FIG. VI.21 – Sigmoïdes associée au seuil s=60 : différentes pentes α

Nous obtenons une nouvelle fonction $Sig_s \circ G$ à interpoler, dont les valeurs sont comprises entre 0 et 1 :

- $Sig_s \circ G(X) > 0.5$ pour les points $X \in D_s$, c'est-à-dire tels que $G(X) > s$,
- $Sig_s \circ G(X) < 0.5$ sinon.

Grâce aux propriétés de la fonction sigmoïde, $Sig_s \circ G$ présente l'avantage de "lisser" les comportements de la fonction dans les zones éloignées de la ligne de niveau associée au seuil s : ainsi, l'interpolation de la nouvelle fonction sera plus facile. Une illustration de ce principe est proposée dans la figure VI.22.

Méthode Pour un seuil s donné,

- créer une *sparse grid* notée g_s en mode adaptatif avec déplacement vers le point nominal qui interpole $Sig_s \circ G$,
- calculer la probabilité de dépasser 0.5 avec une tirage de Monte-Carlo appliqué à g_s : c'est une estimation de la probabilité de dépasser le seuil s avec G.

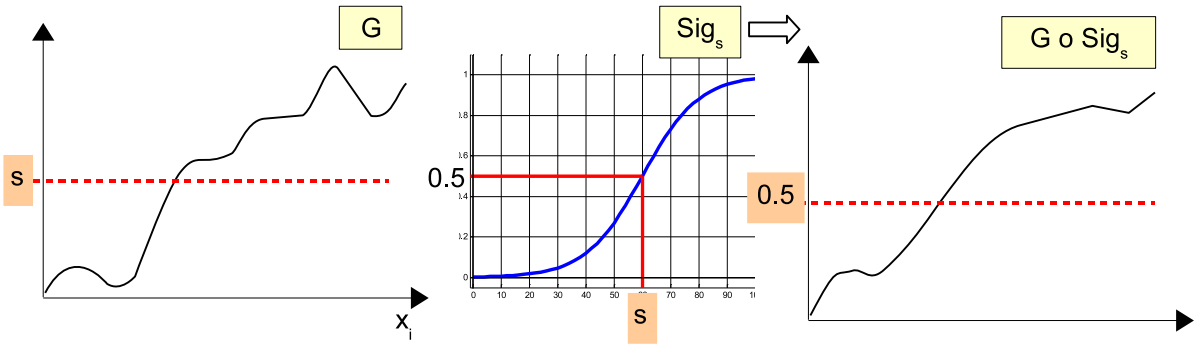


FIG. VI.22 – Seuillage des données : illustration du lissage des données

4.3.3 Point de conception : déplacements successifs

La recherche du point de conception dans ce cas test n'aboutit pas avec la méthode globale utilisée jusqu'ici : le processus d'optimisation ne converge pas.

Ce mauvais résultat s'explique en partie par la non régularité très importante de la fonction de modélisation, mais aussi parce que la dimension du problème a augmenté.

L'idée pour améliorer la recherche du point de conception P^* en grande dimension est

- **restreindre le problème d'optimisation aux paramètres importants**, afin de réduire la dimension du problème.
- **rendre l'approximation plus précise dans la zone qui correspond à la localisation de P^*.**

Avec les données du problème, nous n'avons pas d'indication sur la localisation de P^* : nous pouvons construire une première approximation g_1 en déplaçant les points de grille autour du point nominal, ce point ayant plus de raisons de se trouver proche du point de conception que le centre du domaine de définition.

Nous pouvons alors utiliser g_1 pour calculer la courbe des probabilités, et calculer une première approximation P_1 du point de conception. En ce qui concerne les paramètres importants, nous modifions l'algorithme : l'approximation n'est valable à présent qu'autour du point nominal, or dans le cas test précédent, l'échantillon de points où est évalué le gradient était issu d'un tirage carré latin ne tenant pas compte des lois de probabilités. A présent, nous utiliserons un tirage hypercube latin suivant les lois de probabilités : les points restent bien répartis mais dans un domaine centré sur le point nominal.

Nous détectons donc les paramètres influents au voisinage du point nominal : la notion de paramètre important est ici liée au choix du scénario (elle dépend des lois de probabilités choisies).

Ainsi, nous pouvons calculer une nouvelle approximation en prenant comme centre de la grille P_1, avec une amplitude de 3σ de part et d'autre du centre comme pour le déplacement autour du point nominal.

Nous pouvons également à présent nous restreindre aux paramètres importants.

Algorithme 2 : approximations successives pour une étude de fiabilité

▷ **Données**

Soit une fonction de modélisation $G : \Omega \subset \mathbb{R}^n \longrightarrow \mathbb{R}$ et un scénario associé définissant les lois de probabilité sur le domaine de définition de G.

▷ **Première approximation**

- Construire une ***sparse grid*** g_1 approximation de G **avec déplacement de la grille autour du point nominal** M,
- Utiliser g_1 pour
  - calculer la **courbe des probabilités** de dépasser un seuil,
  - calculer une première **approximation** P_1 **du point de conception**,
  - déterminer les **paramètres importants** et négligeables (notion liée au scénario).

▷ **Approximations successives**

- $g_{k-1} := g_1$, $P_{k-1} := P_1$, $E_k := |G(P_1) - g_1(P_1)|$, $N_k := \|M - P_{k-1}\|$
- Tant que E_k ou N_k sont grands ou la limite du coût de calcul n'est pas atteinte
  - construire une ***sparse grid*** g_k approximation de G **avec déplacement de la grille autour de** P_{k-1} en conservant uniquement les paramètres importants,
  - utiliser g_k pour calculer une approximation P_k du point de conception,
  - $E_k := |G(P_k) - g_k(P_k)|$, $N_k := \|P_k - P_{k-1}\|$

TAB. VI.16 – Algorithme des approximations successives

Le processus peut être poursuivi jusqu'à atteindre

- une convergence par stabilité du point trouvé,
- ou une convergence vers la ligne de niveau associée au seuil s, c'est-à-dire $g(P_k) = G(P_k) = s$
- ou la limite de coût de calcul autorisée.

4.4 Méthode des approximations successives

D'après les améliorations précédentes, nous pouvons proposer la méthode générale suivante, dite des approximations successives, pour l'étude de fiabilité en grande dimension et présentée dans le tableau VI.16.

4.5 Résultats numériques : fonction test

4.5.1 Fonction test analytique G_t

Dans ce cas test à 30 paramètres, la fonction de modélisation semble trop irrégulière : nous l'avons vu pour les probabilités, mais la recherche de points de conception également se heurte à de nombreux minima locaux, et les résultats numériques sont difficiles à interpréter.

Dans la suite, il nous paraît donc préférable de valider les algorithmes avec une fonction test analytique, que nous appellerons aussi G_t, utilisée comme G sous la forme d'une boîte noire.

La forme analytique de G_t est la suivante :

$$\begin{array}{rccl} G_t : & [-10,10]^{30} & \longrightarrow & \mathbb{R}^+ \\ & X & \mapsto & G_t(X) = \left(\left(\sum_{i \in I} -(x_i - 6)^2 \right) + 30 \sqrt{\sum_{j \in J} |x_j - 6|} + b \right)^+ \end{array} \tag{VI.12}$$

avec

- $b = 3,3 \,.\, 10^2$ et $x \mapsto x^+ := max(x, 0)$ la partie positive,
- $I = \{1; 2; 3; 25; 26; 27; 28; 29; 30\}$ les 9 indices de paramètres qui présentent des variations **importantes**, car ils apparaissent sous la forme d'un carré,
- $J = \{4; 5; 6; 15; 16; 17; 18; 19; 20; 21; 22; 23; 24\}$ les 13 indices des paramètres qui présentent des variations **secondaires**, car ils apparaissent dans une racine carrée,
- les 8 paramètres restants sont **négligeables** puisqu'ils n'apparaissent pas dans l'expression.

Scénario En ce qui concerne l'incertitude des paramètres, les lois de probabilités sont toutes gaussiennes, 3 lois différentes sont définies, et sont données dans le tableau VI.17.

Ce scénario présente des caractéristiques communes avec celui du cas test associé à G :

- G_t est à valeurs dans $\mathbb{R}^+$,
- pour $\epsilon \simeq 0$, $\epsilon > 0$, la probabilité $p(G_t(X) > \epsilon)$ est voisine de 0.5, comme constaté dans les résultats précédents avec G, ce qui donne une courbe des probabilités discontinue en 0.

Les tests suivants le montrent, le cas où la fonction de modélisation est valeurs positives avec une courbe des probabilités de dépasser un seuil s associée discontinue en $s = 0$, rend le problème d'approximation difficile (sinon l'approximation *sparse grid* converge quand on augmente le nombre de points). Il nous apparaît donc judicieux de tester la méthode avec ce type de fonction, ce qui explique la formule choisie pour définir G_t.

Paramètre	Intervalle de définition	Loi de probabilité
p_1	[-10 ;10]	N(3,3)
p_2	[-10 ;10]	N(3,3)
p_3	[-10 ;10]	N(3,3)
p_4	[-10 ;10]	N(3,3)
p_5	[-10 ;10]	N(3,3)
p_6	[-10 ;10]	N(3,3)
p_7	[-10 ;10]	N(3,3)
p_8	[-10 ;10]	N(3,3)
p_9	[-10 ;10]	N(3,3)
p_{10}	[-10 ;10]	N(3,3)
p_{11}	[-10 ;10]	N(6,1)
p_{12}	[-10 ;10]	N(6,1)
p_{13}	[-10 ;10]	N(6,1)
p_{14}	[-10 ;10]	N(6,1)
p_{15}	[-10 ;10]	N(6,1)
p_{16}	[-10 ;10]	N(6,1)
p_{17}	[-10 ;10]	N(6,1)
p_{18}	[-10 ;10]	N(6,1)
p_{19}	[-10 ;10]	N(6,1)
p_{20}	[-10 ;10]	N(6,1)
p_{21}	[-10 ;10]	N(-3,0.5)
p_{22}	[-10 ;10]	N(-3,0.5)
p_{23}	[-10 ;10]	N(-3,0.5)
p_{24}	[-10 ;10]	N(-3,0.5)
p_{25}	[-10 ;10]	N(-3,0.5)
p_{26}	[-10 ;10]	N(-3,0.5)
p_{27}	[-10 ;10]	N(-3,0.5)
p_{28}	[-10 ;10]	N(-3,0.5)
p_{29}	[-10 ;10]	N(-3,0.5)
p_{30}	[-10 ;10]	N(-3,0.5)

TAB. VI.17 – Scénario associé à la fonction test G_t

4.5.2 Méthode globale

Nous appliquons la méthode globale utilisée dans le cas test à 9 paramètres (voir section 2.5 page 253) à l'étude du scénario associé à la fonction test G_t.

Cas de convergence : $\widetilde{G_t}$ Si nous modifions le paramètre b de la fonction test, en prenant $b = 5 \,.\, 10^2$, nous obtenons une autre fonction test notée $\widetilde{G_t}$, proche de G_t.

Mais pour $\epsilon \simeq 0,\ \epsilon > 0$, la probabilité $p(G_t(X) > \epsilon)$ est voisine de 1 : la courbe des probabilités est cette fois continue en 0.

Nous conservons le scénario du tableau VI.17.

Si nous construisons des *sparse grids* en mode adaptatif en augmentant le nombre maximum de points de grille : le processus d'approximation converge, ce qui implique que la courbe de probabilités converge aussi (nous n'effectuons pas le reste de l'étude de fiabilité).

Les résultats sont présentés dans la figure VI.23 :

- L'erreur de vérification en fonction du nombre de points maximum souhaités (figure VI.23(a)), est calculée sur un tirage commun de type carré latin de 10^3 points.

  C'est une courbe décroissante, et l'erreur est stabilisée à partir de 3000 points. En effet, si le nombre de points maximum demandé est supérieur à 5000, le nombre de points de grille reste égal à 4300 car le mode adaptatif a convergé (les courbes de probabilités ne sont plus améliorées).

- Sur les figures VI.23(b), VI.23(c) et VI.23(d), les *sparse grids* à 111, 1029 et 3009 points montrent une courbe de probabilités qui converge vers la courbe de référence.

Dans le cas de la fonction test $\widetilde{G_t}$ pour le calcul des probabilités dans le cadre de la méthode globale :

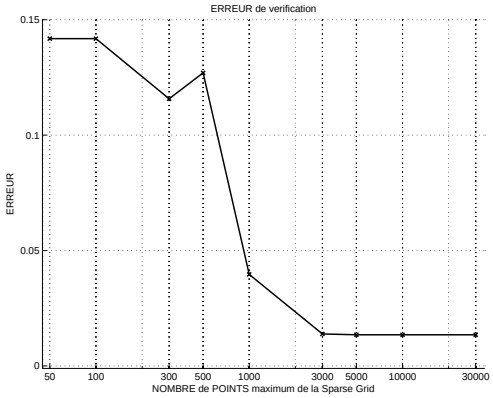
- avec 1000 points de grille, nous obtenons des résultats corrects,

- avec 3000 points de grille, nous obtenons des résultats précis.

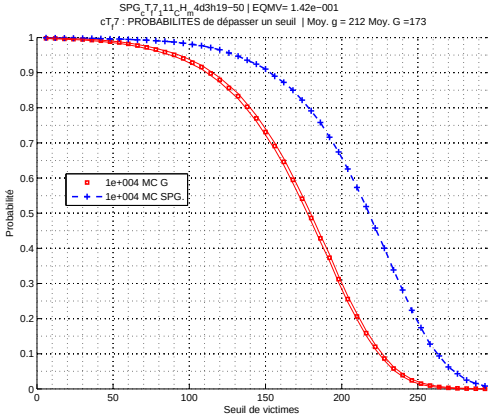
  Le nombre d'évaluations se rapproche du nombre de tirages (10000) de la courbe de référence. Mais contrairement à la méthode de Monte-Carlo appliquée à $\widetilde{G_t}$, avec l'approximation obtenue et avec un coût supplémentaire négligeable, nous pouvons obtenir une étude des paramètres importants et une estimation du point de conception.

Cas de non convergence : G_t Nous construisons ici des *sparse grids* approximations de G_t en mode adaptatif en augmentant le nombre maximum de points de grille : en comparant avec le cas précédent, nous constatons que le processus ne converge plus.

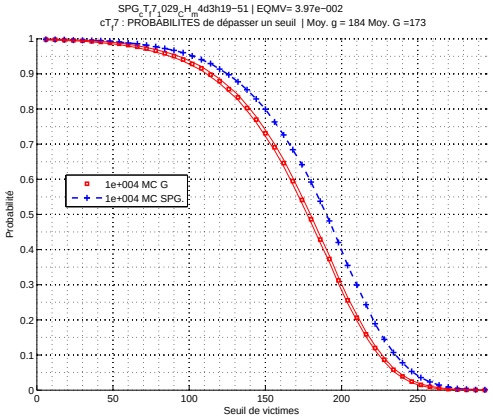
En effet, en examinant les résultats présentés dans la figure VI.24 :



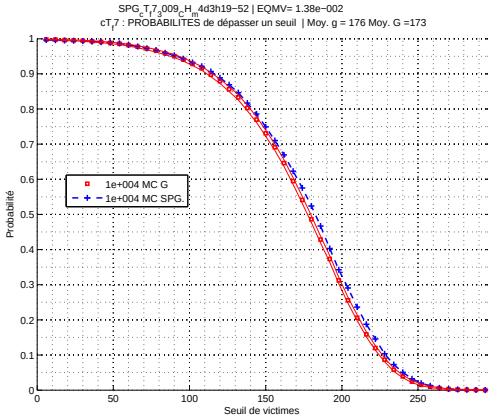
(a) Erreurs de vérification en fonction du nombre de points de grille



(b) *Sparse grid* à 100 points : courbe des probabilités



(c) *Sparse grid* à 1000 points : courbe des probabilités



(d) *Sparse grid* à 3000 points : courbe des probabilités

FIG. VI.23 – *Sparse grids* globales sans déplacement $(\widetilde{G_t})$: courbes des probabilités de dépasser un seuil et erreurs de vérification

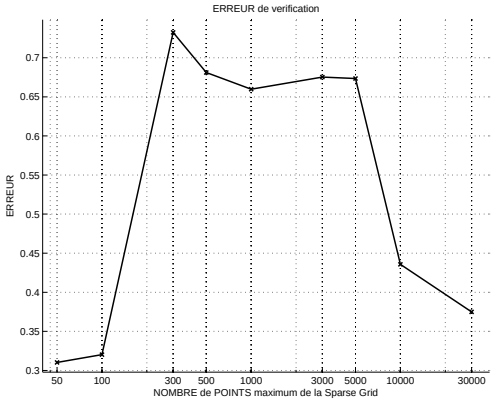
- La courbe de l'erreur de vérification (figure VI.24(d)), calculée comme précédemment, présente une allure différente : l'erreur augmente puis ne décroît qu'à partir de 10000 points, et reste supérieure à l'erreur de départ pour une grille de 50 points.

  L'erreur ne se stabilise pas, et l'approximation ne converge pas.

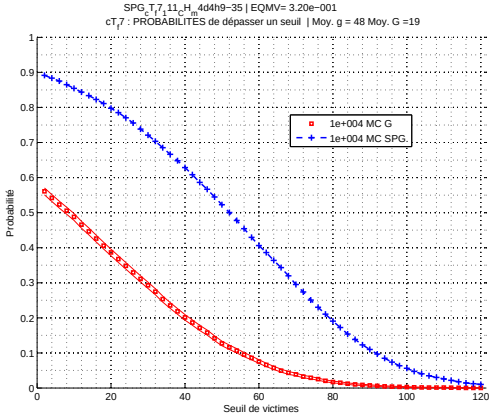
- Sur les figures VI.24(a), VI.24(b) et VI.24(c), les *sparse grids* à 111, 503 et 23825 points montrent une courbe de probabilités qui s'éloigne de la courbe de référence.

  De plus, la dernière *sparse grid* calculée n'a pas utilisé le nombre maximum de points (30000), donc l'erreur estimée par l'algorithme est devenue petite : même en augmentant le nombre de points de grille désiré, l'algorithme adaptatif s'arrêtera au même niveau, et le résultat des probabilités ne sera plus amélioré.

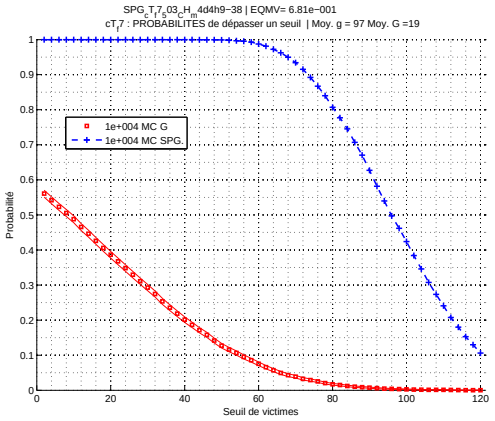
  La courbe de probabilités obtenue avec 100 points reste meilleure que celle de la dernière *sparse grid* à 30000 points.



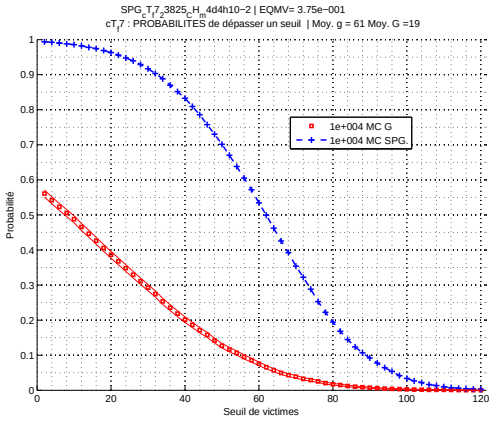
(a) Erreurs de vérification en fonction du nombre de points de grille



(b) *Sparse grid* à 100 points : courbe des probabilités



(c) *Sparse grid* à 500 points : courbe des probabilités



(d) *Sparse grid* à 30000 points : courbe des probabilités

FIG. VI.24 – *Sparse grids* globales **sans déplacement** (G_t) : courbes des probabilités de dépasser un seuil et erreurs de vérification

Les caractéristiques du scénario et de la fonction test G_t, dont la courbe de probabilités présente un saut en 0, rendent le problème d'approximation par *sparse grid* beaucoup plus difficile. L'étude de fiabilité en dimension 30 justifie les améliorations proposées à la méthode globale, que nous testons dans la section suivante.

Sparse grid g_0 **: approximation globale** Nous choisissons la *sparse grid* à 503 points précédente qui présente les meilleurs résultats de probabilités possibles au moindre coût : nous appelons g_0 cette approximation par méthode globale, et nous présentons ici les autres résultats de l'étude de fiabilité de G_t :

- Aucun point de conception n'est trouvé avec g_0 (de même que pour le cas test associé à G) : la convergence échoue pour tous les points initiaux choisis au hasard, ce qui semble correspondre à une approximation avec de nombreux minima locaux.
- Le résultat des paramètres influents par la méthode des gradients est présenté dans la figure VI.25 : tous les paramètres importants (9) et tous les paramètres

secondaires (13) sont détectés correctement, et apparaissent avec exactement le même niveau d'importance, ce qui correspond bien à l'expression analytique.

Le classement par ordre d'importance est présenté dans le tableau VI.18.

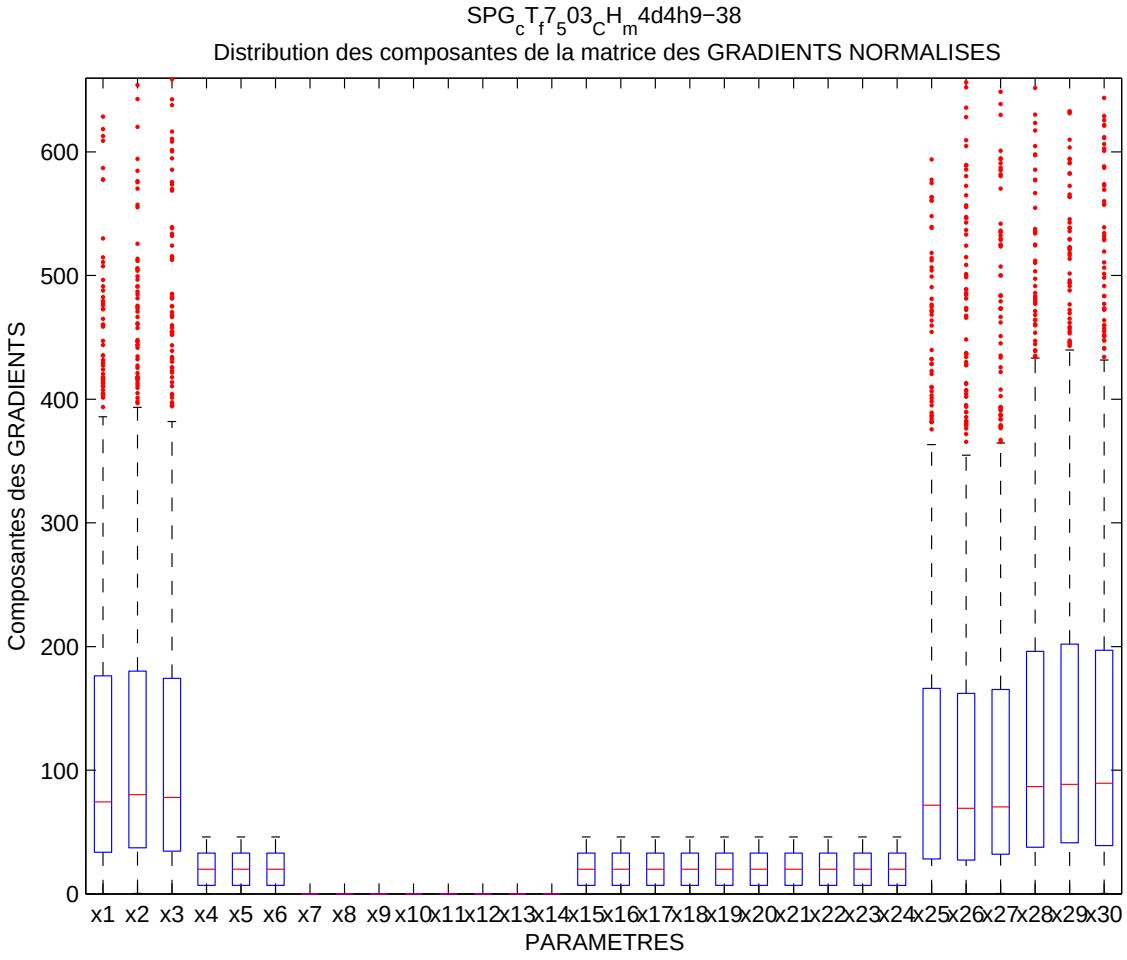


FIG. VI.25 – *Sparse grid* g_0 : distribution des gradients pour classement des paramètres (indépendant du scénario)

Analyse L'approximation g_0 a bien identifié l'importance des paramètres, mais obtient de mauvais résultats concernant les probabilités et le point de conception : la méthode globale montre donc ici ses limites pour une fonction de modélisation à support petit (G_t), de même que pour une fonction trop bruitée (fonction de modélisation G du cas test à 30 paramètres).

La **disposition des points de grille** ne semble **pas appropriée** pour permettre de **détecter** correctement certaines **zones de variations** de G_t, ce que va corriger la méthode des approximations successives.

Paramètres		
principaux	secondaires	négligeables
p_1	p_4	p_7
p_2	p_5	p_8
p_3	p_6	p_9
p_{25}	p_{15}	p_{10}
p_{26}	p_{16}	p_{11}
p_{27}	p_{17}	p_{12}
p_{28}	p_{18}	p_{13}
p_{29}	p_{19}	p_{14}
p_{30}	p_{20}	
	p_{21}	
	p_{22}	
	p_{23}	
	p_{24}	

TAB. VI.18 – *Sparse grid* g_0 : classement des paramètres (indépendant du scénario)

4.5.3 Méthode des approximations successives : application à G_t

Nous appliquons ici la méthode des approximations successives décrite en section 4.4 (voir page page 285) à G_t avec le scénario associé.

Nous efectuons d'abord les mêmes tests de convergence sur le calcul de probabilités que prédédemment avec des *sparse grids* en mode adaptatif avec déplacement vers le point nominal, en augmentant le nombre maximum de points de grille.

Dans le scénario associé à la fonction test G_t, les dix premiers paramètres suivent une loi gaussienne $N(m = 3, \sigma = 3)$ sur un intervalle de définition $[-10, 10]$: quand nous centrons la grille sur le point nominal avec un écart de 3 σ, l'intervalle devient $[-6, 12]$ et nous sommes obligés de tronquer la grille sur l'intervalle $[-6, 10]$.

Nous pouvons ainsi d'abord examiner l'amélioration éventuelle apportée par le changement de variable par homothétie portant sur les points de grille intérieurs de manière à avoir l'intersection des axes centraux sur le point nominal (voir section 4.3.1 page 280).

Probabilités : déplacement sans transformation des points intérieurs Si nous effectuons d'abord le déplacement vers le point nominal mais sans tranformer les points de grille intérieurs par homothétie, nous obtenons les résultats de la figure VI.26 :

- L'erreur de vérification (figure VI.26(a)) reste stable : si on fait un zoom elle décroît très légèrement jusqu'au nombre 10000 de points de grille maximum, puis pour 30000 points elle devient très grande.

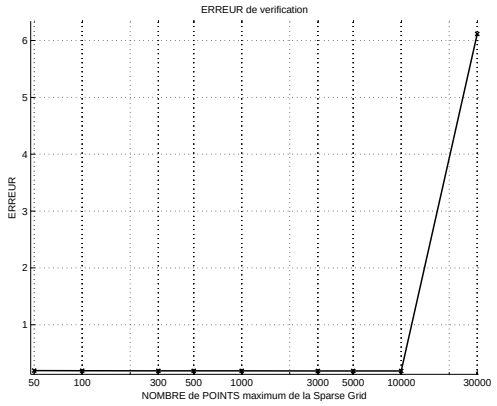
  L'approximation utilise ici tous les points de grille souhaités.

- Les *sparse grids* à 111, 10031 et 30003 points (figures VI.26(b) à VI.26(d)), et celles avec le nombre de points intermédiaires non présentées ici, montrent une courbe de probabilités qui reste la même jusqu'à la dernière *sparse grid* à 30000 points.

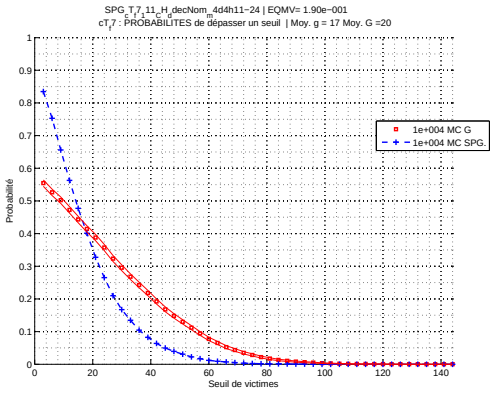
  Pour la *sparse grid* à 30003 points, l'écart avec la courbe de référence se réduit nettement pour les seuils supérieurs à $s = 30$ mais augmente pour les seuils inférieurs à $s = 30$. L'erreur de vérification grande doit correspondre ainsi au nombreux points X du domaine pour lesquels $G_t(X)$ est égal à zéro ou est petit.

Analyse Nous constatons l'amélioration apportée au calcul des probabilités par le simple décalage vers le point nominal, mais les résultats ne sont pas encore assez précis. De plus le processus ne converge pas au vu du comportement de l'erreur de vérification.

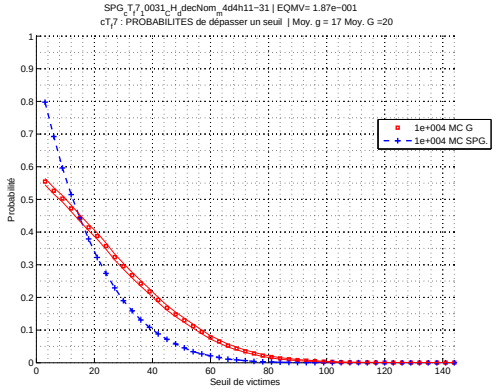
Probabilités : déplacement avec transformation des points intérieurs Cette fois, nous effectuons à la fois le déplacement vers le point nominal et la tranformation des points de grille intérieurs par homothétie, nous obtenons les résultats de la figure VI.27 :



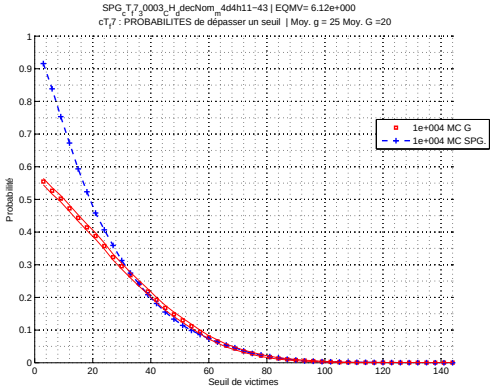
(a) Erreurs de vérification en fonction du nombre de points de grille



(b) *Sparse grid* à 100 points : courbe des probabilités



(c) *Sparse grid* à 1000 points : courbe des probabilités



(d) *Sparse grid* à 3000 points : courbe des probabilités

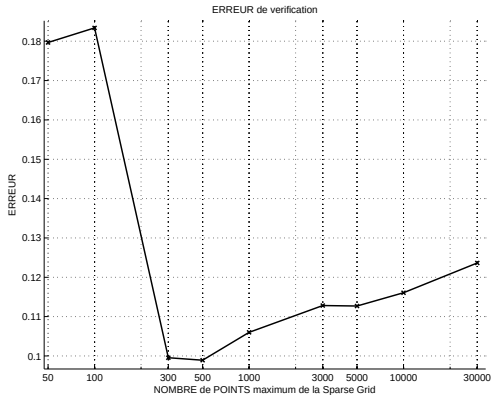
FIG. VI.26 – *Sparse grids* **avec déplacement** vers le point nominal, **sans transformation** des points intérieurs : courbes des probabilités de dépasser un seuil et erreurs de vérification

- L'erreur de vérification (figure VI.27(a)) décroît pour atteindre un minimum pour 300 et 500 points, puis augmente à nouveau.
- Les *sparse grids* à 111, 303 et 3033 points (figures VI.27(b) à VI.27(d)) montrent l'évolution de la courbe de probabilités.

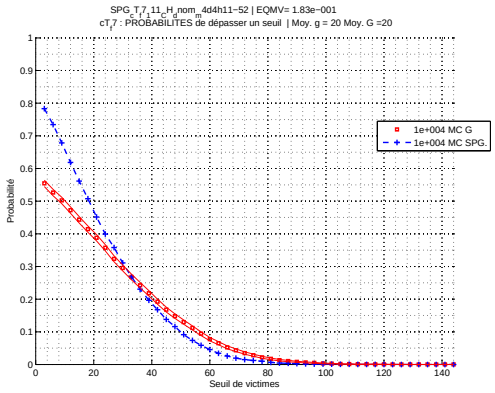
  Elle commence par croiser la courbe de référence, avec un écart important sur les seuils bas, puis avec 300 points l'écart est nettement réduit pour les seuils supérieurs à $s = 30$ et légèrement réduit pour les seuils bas.

  Puis à partir de 3000 points de grille, l'écart se réduit sur les seuils proches de zéro mais mais augmente pour les autres. Si on pousse jusqu'à 30057 points (non affiché), la courbe reste sensiblement la même.

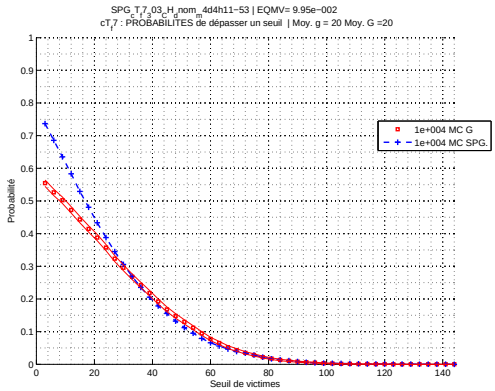
Analyse Le **changement de variable** sur les points de grille intérieurs apporte une précision supplémentaire sensible dans le cas d'un scénario pour lequel l'écart type de certaines variables importantes suivant une loi $N(m, \sigma)$ implique que le support $[m - 3\ \sigma, m + 3\ \sigma]$ n'est pas inclus dans leur intervalle de définition.



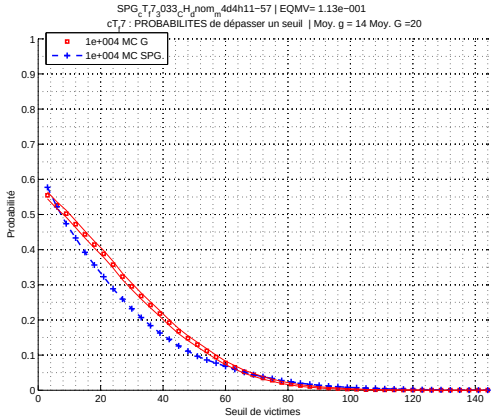
(a)



(b) *Sparse grid* à 100 points : courbe des probabilités



(c) *Sparse grid* à 300 points : courbe des probabilités



(d) *Sparse grid* à 3000 points : courbe des probabilités

FIG. VI.27 – *Sparse grids* **avec déplacement** vers le point nominal (**avec transformation** des points intérieurs) : courbes des probabilités de dépasser un seuil et erreurs de vérification

Même si on peut travailler avec des scénarios dans lesquels ce cas de figure n'apparaît pas (et le changement de variable par homothétie devient inutile), ces résultats font apparaître la **nécéssité de placer les points de grille dans les zones d'intérêt** pour le calcul.

Sparse grid g_1 **: première approximation** Dans le cas de la fonction test G_t et le calcul des probabilités, l'approximation à **300 points** avec déplacement et homothétie des points de grille nous donne un résultat correct (voir figure VI.27(c)) : nous appelons g_1 cette *sparse grid*, qui correspond à la **première approximation de la méthode des approximations successives**.

Il faut 3000 points de grille pour obtenir une amélioration significative sur les résultats de probabilité : or avec g_1 nous allons obtenir les mêmes résultats que pour 3000 points concernant les paramètres importants et le point de conception.

Point de conception Nous appliquons la recherche des points de conception P^* à g_1 : nous présentons la méthode avec un seuil $s = 60$, car dans ce cas, la première approximation de P^* doit être améliorée, cette condition est requise pour tester les approximations successives (pour d'autres valeurs de seuil, g_1 donne directement un point très proche de P^*).

Les résultats sont donnés dans le tableau VI.19.

Analyse Cette fois, le processus d'optimisation converge, nous obtenons un seul point appelé P_1 : ses coordonnées sont différentes des moyennes des lois gaussiennes pour les 9 paramètres principaux, mais également pour certains paramètres secondaires.

Le point obtenu n'est cependant pas au bord du domaine de défaillance associé à G_t et au seuil $s = 60$ puisque $G_t(P_1) = 56.30$.

Améliorations successives Pour obtenir un résultat plus précis nous construisons une autre *sparse grid* g_2 centrée sur P_1, puis nous relançons la recherche du point de conception : nous avons obtenus de bons résultats avec 100 points de grille au maximum.

Les résultats sont présentés dans le tableau VI.20 :

- Nous obtenons un seul point appelé P_2, qui est beaucoup plus proche du bord du domaine de défaillance puisque cette fois $G_t(P_2) = 58.71 \simeq 60$.
- Le résidu a un peu diminué et vaut 1.98, ce qui signifie que la deuxième approximation P_2 du point de conception est un peu plus proche du point nominal que P_1.
- Les coordonnées de P_2 diffèrent légèrement de celles de P_1 uniquement pour les paramètres qui avaient déjà pris une valeur différente de la moyenne.

Si nous relançons le processus avec une nouvelle *sparse grid* g_3 à 100 points de grille centrée sur P_2, les coordonnées sont encore légèrement modifiées et nous obtenons un unique point P_3 qui vérifie $G_t(P_3) = 60.09$ avec un résidu égal à 2.11 (résultats non affichés).

Une dernière approximation g_4 à 100 points de grille trouve un point de conception P_4 qui est au bord du domaine de défaillance avec $G_t(P_4) = 60.01$, et les coordonnées de P_4 et P_3 sont quasiment les mêmes (résultats non affichés).

Analyse Nous pouvons considérer que dans le cas considéré la méthode a convergé en P_3, et nous trouvons déjà avec P_2 une bonne approximation du point de conception.

De plus le coût supplémentaire exigé par les approximations successives apparaît très raisonnable, puisque dans le cas présenté, nous pouvons nous contenter d'un nombre réduit à 100 de points de grille supplémentaires avec g_2.

```
-----------------------------------------------------------------
  Résultats du calcul des points de CONCEPTION:
-----------------------------------------------------------------
  Seuil de victimes:  60
-----------------------------------------------------------------
  Résidus fonction coût:
         2.04     2.04     2.04     2.04     2.04
-----------------------------------------------------------------
  COORDONNEES des points de CONCEPTION P*:
.................................................................
           P*1      P*2      P*3      P*4      P*5
p_1       3.94     3.94     3.94     3.94     3.94
p_2       3.26     3.26     3.26     3.26     3.26
p_3       3.26     3.26     3.26     3.26     3.26
p_4       2.78     2.78     2.78     2.78     2.78
p_5       2.78     2.78     2.78     2.78     2.78
p_6       2.78     2.78     2.78     2.78     2.78
p_7       3.00     3.00     3.00     3.00     3.00
p_8       3.00     3.00     3.00     3.00     3.00
p_9       3.00     3.00     3.00     3.00     3.00
p_10      3.00     3.00     3.00     3.00     3.00
p_11      6.00     6.00     6.00     6.00     6.00
p_12      6.00     6.00     6.00     6.00     6.00
p_13      6.00     6.00     6.00     6.00     6.00
p_14      6.00     6.00     6.00     6.00     6.00
p_15      6.00     6.00     6.00     6.00     6.00
p_16      6.00     6.00     6.00     6.00     6.00
p_17      6.00     6.00     6.00     6.00     6.00
p_18      6.00     6.00     6.00     6.00     6.00
p_19      6.00     6.00     6.00     6.00     6.00
p_20      6.00     6.00     6.00     6.00     6.00
p_21     -3.03    -3.03    -3.03    -3.03    -3.03
p_22     -3.03    -3.03    -3.03    -3.03    -3.03
p_23     -3.03    -3.03    -3.03    -3.03    -3.03
p_24     -3.03    -3.03    -3.03    -3.03    -3.03
p_25     -2.73    -2.73    -2.73    -2.73    -2.73
p_26     -2.73    -2.73    -2.73    -2.73    -2.73
p_27     -2.73    -2.73    -2.73    -2.73    -2.73
p_28     -2.73    -2.73    -2.73    -2.73    -2.73
p_29     -2.73    -2.73    -2.73    -2.73    -2.73
p_30     -2.73    -2.73    -2.73    -2.73    -2.73
.................................................................
g(P*)=   60       60       60       60       60
G(P*)=   56.30    56.30    56.30    56.30    56.30
-----------------------------------------------------------------
 Succès optim.:     100
```

TAB. VI.19 – *Sparse grid* g_1 : points de conception pour s=60

```
-----------------------------------------------------------------
  Résultats du calcul des points de CONCEPTION:
-----------------------------------------------------------------
  Seuil de victimes:  60
-----------------------------------------------------------------
  Résidus fonction coût:
         1.98     1.98     1.98     1.98     1.98
-----------------------------------------------------------------
  COORDONNEES des points de CONCEPTION P*:
.................................................................
           P*1      P*2      P*3      P*4      P*5
p_1       3.95     3.95     3.95     3.95     3.95
p_2       3.98     3.98     3.98     3.98     3.98
p_3       3.98     3.98     3.98     3.98     3.98
p_4       2.77     2.77     2.77     2.77     2.77
p_5       2.77     2.77     2.77     2.77     2.77
p_6       2.77     2.77     2.77     2.77     2.77
p_7       3.00     3.00     3.00     3.00     3.00
p_8       3.00     3.00     3.00     3.00     3.00
p_9       3.00     3.00     3.00     3.00     3.00
p_10      3.00     3.00     3.00     3.00     3.00
p_11      6.00     6.00     6.00     6.00     6.00
p_12      6.00     6.00     6.00     6.00     6.00
p_13      6.00     6.00     6.00     6.00     6.00
p_14      6.00     6.00     6.00     6.00     6.00
p_15      6.00     6.00     6.00     6.00     6.00
p_16      6.00     6.00     6.00     6.00     6.00
p_17      6.00     6.00     6.00     6.00     6.00
p_18      6.00     6.00     6.00     6.00     6.00
p_19      6.00     6.00     6.00     6.00     6.00
p_20      6.00     6.00     6.00     6.00     6.00
p_21     -3.03    -3.03    -3.03    -3.03    -3.03
p_22     -3.03    -3.03    -3.03    -3.03    -3.03
p_23     -3.03    -3.03    -3.03    -3.03    -3.03
p_24     -3.03    -3.03    -3.03    -3.03    -3.03
p_25     -2.78    -2.78    -2.78    -2.78    -2.78
p_26     -2.78    -2.78    -2.78    -2.78    -2.78
p_27     -2.78    -2.78    -2.78    -2.78    -2.78
p_28     -2.76    -2.76    -2.76    -2.76    -2.76
p_29     -2.78    -2.78    -2.78    -2.78    -2.78
p_30     -2.76    -2.76    -2.76    -2.76    -2.76
.................................................................
g(P*)=   60       60       60       60       60
G(P*)=   58.71    58.71    58.71    58.71    58.71
-----------------------------------------------------------------
 Succès optim.:     100
```

TAB. VI.20 – *Sparse grid* g_2 : points de conception pour s=60

Paramètres					
principaux		secondaires			négligeables
p_1	p_{25}	p_4	p_{15}	p_{21}	p_7
p_2	p_{26}	p_5	p_{16}	p_{22}	p_8
p_3	p_{27}	p_6	p_{17}	p_{23}	p_9
	p_{28}		p_{18}	p_{24}	p_{10}
	p_{29}		p_{19}		p_{11}
	p_{30}		p_{20}		p_{12}
					p_{13}
					p_{14}

TAB. VI.21 – *Sparse grid* g_1 : classement des paramètres au voisinage du point nominal, par ordre d'importance de gauche à droite

Paramètres importants La méthode des gradients appliquée à g_1 est illustrée dans la figure VI.28.

Les paramètres principaux et secondaires apparaissent comme dans le cas sans déplacement avec g_0, mais avec des niveaux d'importance différents cette fois, puisque l'approximation g_1 n'est pas valable sur tout le domaine (mais autour du point nominal).

Le classement par ordre d'importance est présenté dans le tableau VI.21, avec des sous-catégories par ordre d'importance de gauche à droite.

Ainsi nous pouvons affiner le résultat par rapport à celui obtenu avec g_0, en disant ici : au voisinage du point le plus probable,

- les paramètres les plus influents sont d'abord $\{p_1, p_2, p_3\}$, puis $\{p_{25}, p_{26}, p_{27}, p_{28}, p_{29}, p_{30}\}$,
- les paramètres secondaires sont d'abord $\{p_4, p_5, p_6\}$, puis $\{p_{15}, p_{16}, p_{17}, p_{18}, p_{19}, p_{20}\}$, et enfin $\{p_{21}, p_{22}, p_{23}, p_{24}\}$,
- les paramètres négligeables sont $\{p_7, p_8, p_9, p_{10}, p_{11}, p_{12}, p_{13}, p_{14}\}$.

Analyse L'information sur l'importance des paramètres est ici complémentaire, et permet une interprétation différente du **classement des paramètres** (comparer les tableaux VI.18 page 292 et VI.21) si l'utilisateur s'intéresse à leur **influence au voisinage du point nominal** plutôt que de manière indépendante du scénario.

Les niveaux atteints par le mode adaptatif permettent de détecter les paramètres principaux, mais pas de les classer entre eux autour du point nominal, et ne permettent pas de détecter tous les paramètres secondaires : la méthode de distribution des gradients est plus riche en informations.

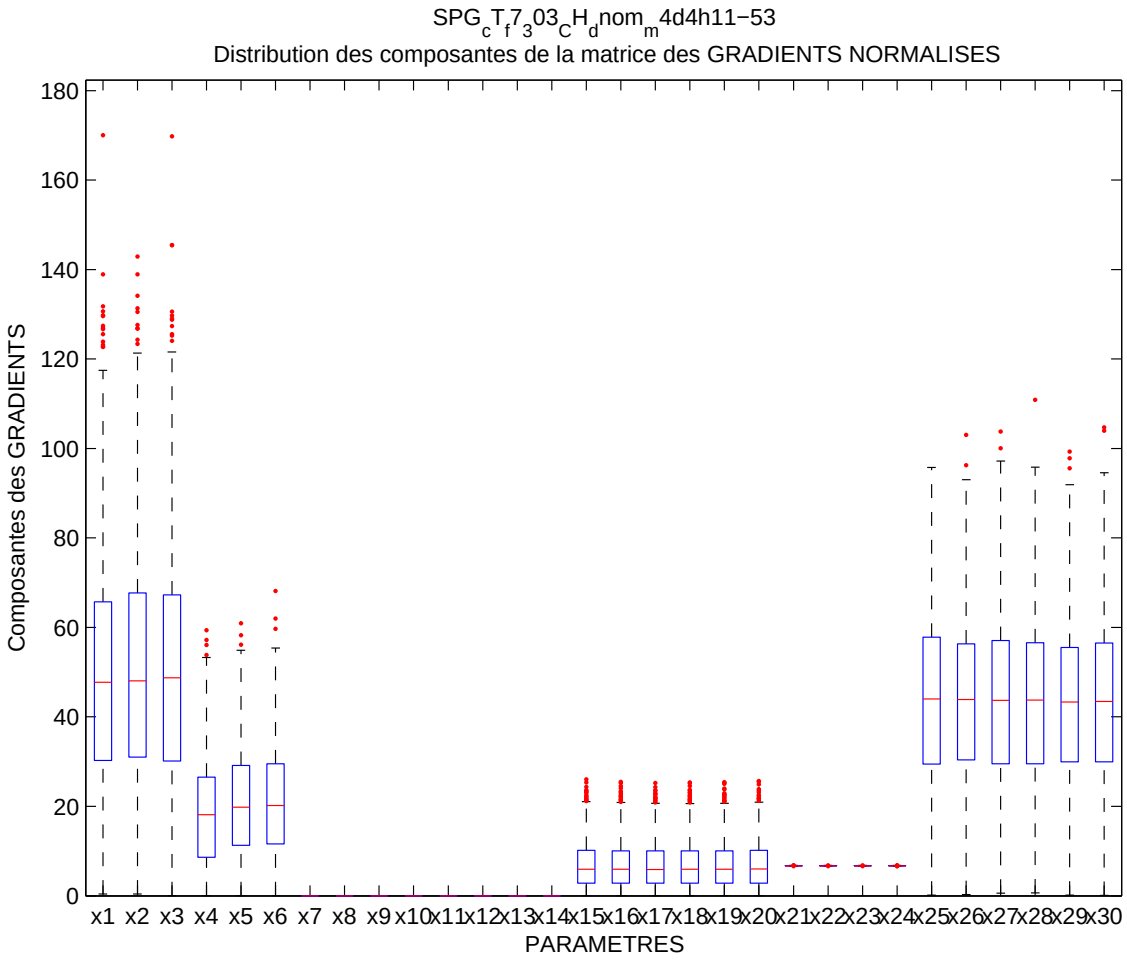


FIG. VI.28 – *Sparse grid* g_1 : distribution des gradients pour classement des paramètres autour du point nominal

Probabilités : seuillage des données

Affichage La courbe de probabilités obtenue avec g_1 n'est pas très proche de la courbe de référence pour tous les seuils, nous testons ici la méthode du seuillage des données, afin d'améliorer ces résultats.

Nous allons effectuer le calcul pour différentes valeurs de la pente α de la sigmoïde de seuillage (voir section 4.3.2 page 282) pour étudier l'influence de ce coefficient.

Nous calculons aussi une erreur de vérification pour chaque *sparse grid* sur un même tirage de type hypercube latin (tenant compte des lois de probabilité) de 1000 points.

Sur les graphiques de probabilités suivants apparaissent donc

- la pente α de la sigmoïde en abcisses :

  pour des valeurs inférieures à 5 la sigmoïde est raide et pour des valeurs supérieures à 50 la sigmoïde est plutôt de type linéaire,

- les probabilités en ordonnées, avec :

  - le résultat de référence : la méthode de Monte-Carlo avec un échantillon de 10^4 points représenté par un symbole "$\square$" et l'intervalle de confiance à 95% estimé en pointillés,
  - le résultat de la méthode de Monte-Carlo avec 100 points, afin de comparer avec la méthode du seuillage pour lequel nous utiliserons environ ce nombre d'évaluations.

    Comme le nombre de points est ici faible, la méthode présente ici des variations de résultat importantes et l'estimation de l'intervalle de confiance utilisée pour 10^4 points n'est plus fiable.

    Nous calculons alors un intervalle de confiance à 95% en appliquant 10^4 fois la méthode et en étudiant la répartition des résultats : il est représenté avec des pointillés fins.
  - le résultat obtenu avec la *sparse grid* g_1 (méthode globale), représenté par le symbole "$\circ$",
  - les résultats du seuillage représentés par des symboles "$\triangle$".

Résultats Nous avons choisi trois seuils correspondant à trois ordres de grandeurs de probabilités différents :

- $s = 6$ correspondant à une probabilité de l'ordre de 0.5,
- $s = 60$ correspondant à une probabilité de l'ordre de 0.07,
- $s = 90$ correspondant à une probabilité de l'ordre de 0.007.

Après chaque seuillage de données, nous calculons une *sparse grid* avec déplacement vers le point nominal (sinon les résultats ne sont pas satisfaisants), en donnant un nombre de points de grille maximum de 50 ou 100 : ici une approximation différente est calculée pour chaque seuil choisi et ne peut être utilisée que pour l'estimation de

la probabilité associée à ce seuil, la méthode ne doit donc utiliser qu'un nombre très réduit de points.

Nous n'affichons pas l'erreur de vérification, car les probabilités issues du seuillage ne dépendent pas directement de la qualité de l'approximation, qui doit simplement bien séparer les valeurs autour de 0.5 pour les données seuillées. Les meilleurs résultats ne correspondent effectivement pas aux erreurs les plus petites.

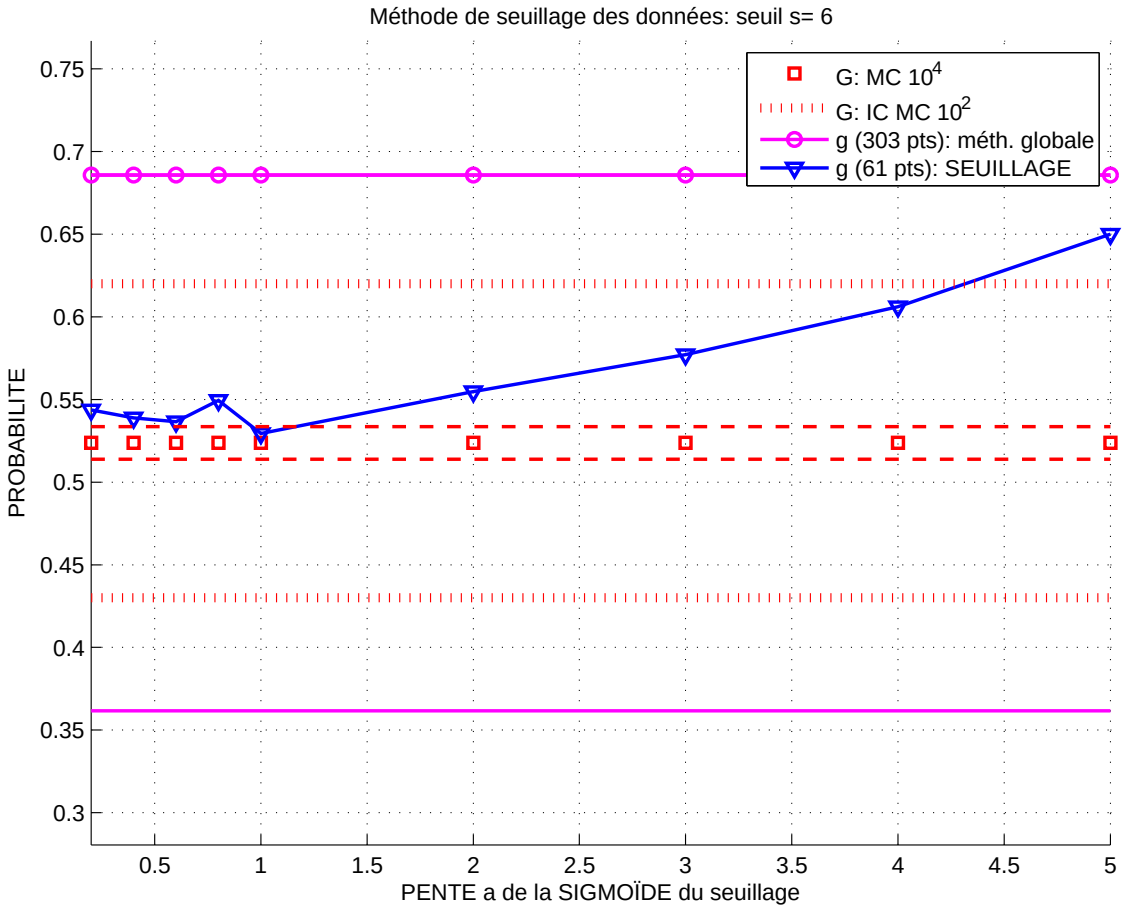


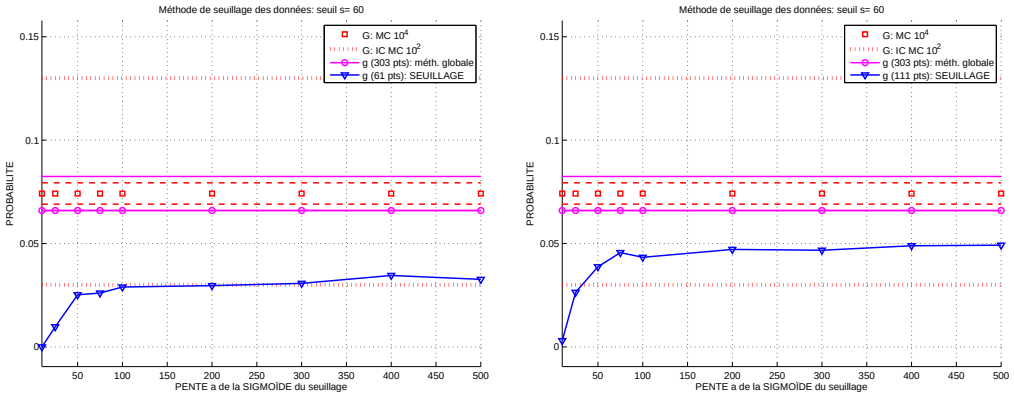
FIG. VI.29 – *Sparse grids* à 61 points avec seuillage : probabilités de dépasser le seuil $s = 6$ en fonction de la pente de la sigmoïde

En ce qui concerne le seuil $s = 6$, avec 61 points nous obtenons des probabilités, présentées dans la figure VI.29, de la même précision que le résultat de référence obtenu avec 10^4 points, pour des valeurs de α inférieures à un.

La courbe des probabilités obtenue pour g_1 avec 300 points et la méthode globale présentait un écart important avec la référence pour les seuils bas : dans ce cas $s = 6$, le gain en coût et en précision est significatif.

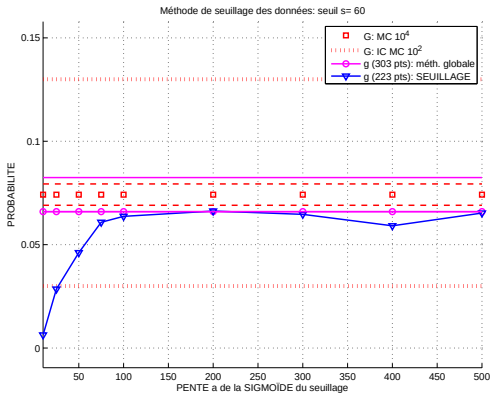
Pour les seuils supérieurs à $s = 30$, les probabilités obtenues avec g_1 sont très précises : nous ne pouvons pas espérer les améliorer, mais nous pouvons essayer de trouver des résultats similaires avec un nombre de points inférieur.

Pour le seuil $s = 60$, la probabilité calculée avec 61 points (figure VI.30(a)) se rapproche de celle de référence pour des valeurs de seuil beaucoup plus grandes, à



(a) *Sparse grids* à 61 points

(b) *Sparse grids* à 111 points



(c) *Sparse grids* à 223 points

FIG. VI.30 – *Sparse grids* avec seuillage : probabilités de dépasser le seuil $s = 60$ en fonction de la pente de la sigmoïde

partir de $\alpha = 50$: elle est cependant du même ordre que la borne inférieure de l'intervalle de confiance associé au tirage de Monte-Carlo à 100 points, et pas aussi précise que la probabilité issue de g_1.

Avec un nombre légèrement supérieur de 111 points (figure VI.30(b)), les probabilités calculées se rapprochent nettement de la référence, pour un coût inférieur à celui de g_1 : le résultat est plus précis que celui de la méthode de Monte-Carlo avec 100 points.

Si nous augmentons encore le nombre de points, avec 223 points (figure VI.30(c)), le résultat du seuillage est du même ordre de précision que pour g_1 pour un coût un peu inférieur.

Pour le seuil $s = 90$ (figure VI.31), le résultat des *sparse grids* avec seuillage à 111 points, pour des valeurs de α supérieures à 100, sont encore ici plus précises que la méthode de Monte-Carlo pour un coût similaire, et presque autant que g_1 pour un coût inférieur.

Analyse Nous constatons des comportements différents par rapport au coefficient de la pente α de la sigmoïde de seuillage des données. La méthode est satisfaisante

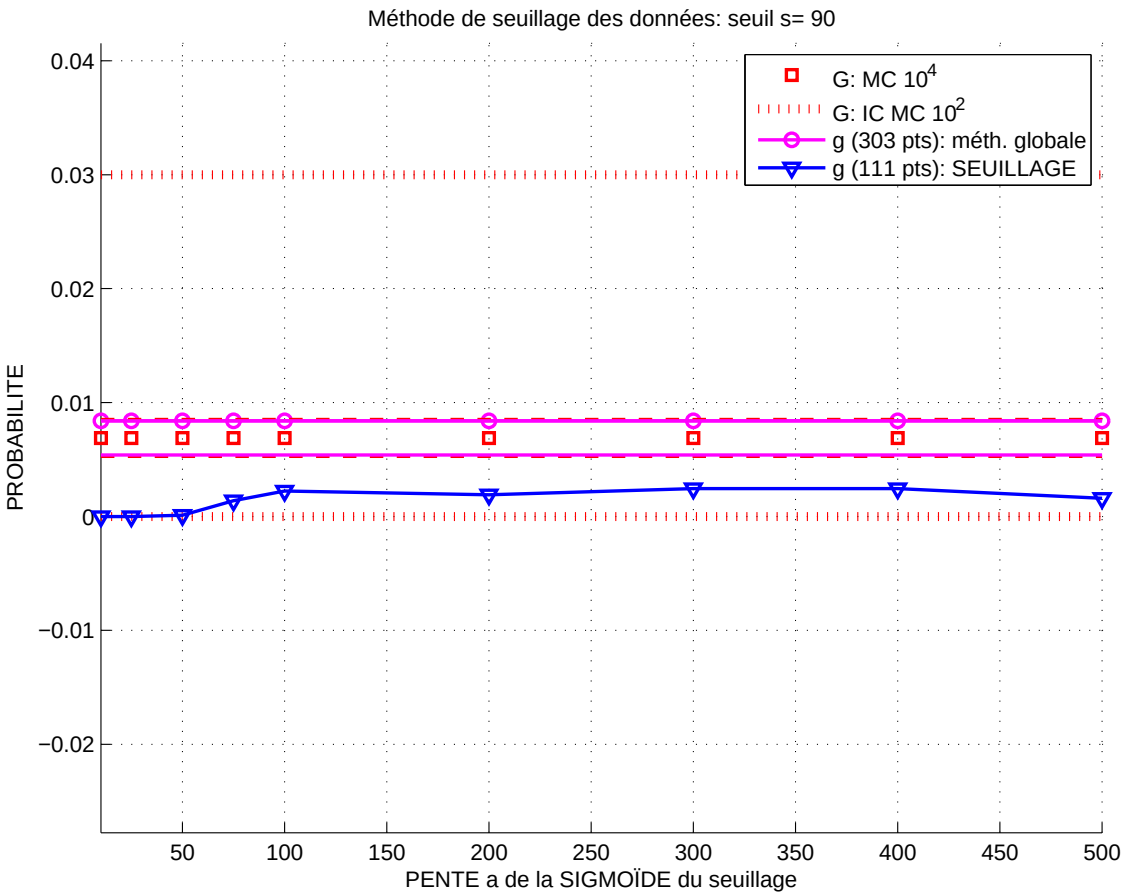


FIG. VI.31 – *Sparse grids* à 111 points avec seuillage : probabilités de dépasser le seuil $s = 90$

- pour α compris entre 0.1 et 5, c'est-à-dire des sigmoïdes raides, quand le seuil choisi est bas ($s = 6$),
- pour α supérieur à 50, c'est-à-dire des sigmoïdes de type linéaire, quand le seuil choisi est haut ($s = 60$ et $s = 90$).

Pour ces ordres de grandeur de seuil différents, nous avons vu que la méthode du **seuillage des données** donne

- soit un résultat beaucoup **plus précis que la méthode globale pour un coût inférieur** quand celle-ci n'est pas proche de la référence,
- soit un **résultat presque aussi précis** pour un coût légèrement inférieur.

Comme les *sparse grids* ne semblent pas très adaptées à l'approximation de fonctions raides (voir [Kli05]), il serait intéressant de tester la méthode de seuillage des données avec d'autres techniques d'approximation plus adaptées à ce cas, ou avec des techniques de classification.

Il faudrait aussi vérifier si le gain en coût de calcul apporté par le seuillage augmente avec la dimension du problème, ce qui en ferait une méthode très intéressante.

4.6 Conclusion

Pour le cas test en dimension $n = 30$ avec la fonction test G_t à valeurs positives de façon à ce que la courbe des probabilités de dépasser un seuil s soit non continue en $s = 0$ (cas difficile) :

- La méthode d'**approximation globale** est mise en défaut : la *sparse grid* g_0 ne donne pas de résultats satisfaisants pour les probabilités et ne trouve pas de point de conception (voir section 4.5.2 page 290).

  Elle **détecte correctement les directions importantes** en appliquant la méthode de **distribution des gradients** (voir section 2.7.1 page 254).

  Il semble que la disposition fixe des points de grille, malgré le mode adaptatif qui place plus de points dans les directions difficiles à apprendre, ne soit pas adaptée à tous les types de fonctions à approcher.

  De plus, quand on augmente le nombre de points de grille, les résultats ne sont pas améliorés, voire même dégradés pour les trois problèmes à résoudre (voir section 4.2.1 page 277) : l'approximation ne converge pas.

  Comme les *sparse grids* sont basées sur l'interpolation, la **non-convergence** doit provenir des **instabilités numériques** propres à cette méthode (voir section 1.5.2 page 198).

- Nous avons alors effectué l'étude en appliquant les améliorations basées sur la **méthode des approximations successives** (voir section 4.4 page 285).

  - La **première approximation** *sparse grid* g_1 à 300 points de grille est obtenue en plaçant l'intersection des axes centraux de la grille sur le point nominal (translation de la grille puis homothétie des points intérieurs, voir section 4.3.1 page 280).

    La **courbe des probabilités** obtenue est à présent **correcte**, et une approximation P_1 du point de conception est trouvé, dont l'évaluation par G_t est assez proche du bord du domaine de défaillance (voir section 4.5.3 page 295).

    Un classement des paramètres importants est effectué, il est légèrement différent du classement déduit de la formule analytique de G_t car il est valable au voisinage du point nominal.

    On peut noter qu'avec le décalage vers le point nominal mais sans l'homothétie sur les points intérieurs, les résultats sur les probabilités sont améliorés mais ne sont pas encore satisfaisants (voir section 4.5.3 page 293) : la **disposition des données de construction**, représentée ici par les axes centraux de la grille, est donc **primordiale** pour obtenir des résultats précis.

    Grâce aux **améliorations** apportées, nous avons donc une **étude de fiabilité correcte** pour un **coût de 300 points en dimension 30**, pour une fonction G_t **non bruitée mais difficile à approcher**.

  - La deuxième approximation *sparse grid* g_2 à 100 points de grille est obtenue par déplacement de la grille vers le point P_1.

    Elle permet pour un **coût supplémentaire faible** de trouver une approximation P_2 du point de conception qui améliore le résultat. Avec des

déplacement supplémentaires et les *sparse grids* g_3 et g_4 à 100 points associées, le **processus converge vers le point de conception**.

Conclusion

Bilan

Nous avons effectué une analyse de fiabilité et de sensibilité pour plusieurs fonctions de modélisation de type boîte noire, avec :

- calcul des probabilités de dépasser un seuil,
- recherche du point de conception,
- détermination des paramètres importants.

Approximation globale

La première méthode consiste à construire une **approximation globale** du modèle initial, avec un objectif de coût total réduit, et de l'utiliser pour effectuer l'analyse : le principal avantage est de disposer du **gradient**.

Les résultats sont obtenus

- pour les probabilités avec une **méthode de Monte-Carlo**,
- pour le point de conception avec une **méthode classique de descente avec pénalisation** de la contrainte,
- pour les paramètres importants avec la **représentation statistique d'une distribution des gradients** évalués sur un tirage de type **carré latin**.

Pour le cas test avec **9 paramètres incertains** et la fonction de modélisation du CEG simulant un accident chimique, deux techniques d'approximation ont été utilisées :

- les **réseaux de neurones**, basés sur une disposition des **données de construction** respectant les distributions de probabilités des paramètres incertains par tirage **hypercube latin** et sur un apprentissage par moindres carrés utilisant des méthodes de **régularisation**.
- les *sparse grids*, qui consistent en une interpolation sur grille éparse à **caractère hiérarchique**, basées sur un **algorithme adaptatif** qui place les points de grille dans les directions difficiles à apprendre.

Les deux techniques ont donné des résultats similaires et satisfaisants :

- avec un coût de 500 points pour le réseau de neurones,
- avec un coût de **100 points** pour la *sparse grid*, mais la recherche du point de conception n'aboutit pas dans ce cas là.

La technique des ***sparse grids*** s'est avérée **plus performante** dans le cadre d'une utilisation en grande dimension grâce à la capacité de son algorithme adaptatif à concentrer les évaluations dans des directions privilégiées.

Approximations successives et seuillage des données

Pour le cas test avec **30 paramètres incertains** et une fonction de modélisation non bruitée mais difficile à approcher, la méthode d'approximation globale a montré ses limites.

Nous avons alors appliqué une méthode d'**approximations successives** aux *sparse grids*, qui **dispose les données de construction dans la zone d'intérêt** (par déplacement des points de grille et éventuellement changement de variable).

Cette méthode a permis d'obtenir des **résultats satisfaisants** avec un coût de **300 points** pour les **probabilités** et les **directions importantes**. Avec un **coût supplémentaire de 100 points** par nouvelle approximation, le **processus converge** en deux ou trois itérations **vers le point de conception**.

Pour améliorer la précision du calcul des **probabilités** pour certains seuils, nous avons appliqué une méthode de **seuillage des données** : on construit une **approximation associée à un seuil donné**, en **séparant les données de construction par application d'une sigmoïde** entre celles qui appartiennent au domaine de défaillance et les autres.

L'utilisation de cette méthode avec les *sparse grids* s'est avérée **pertinente**, avec des résultats pour les probabilités améliorés pour certains seuils, avec un coût réduit d'une centaine de points pour chaque approximation.

La qualité de la réponse dépend cependant fortement de la pente de la sigmoïde appliquée, et la validation du résultat est délicate.

Création du logiciel CEGAP

Cette étude a donné lieu à la création d'un logiciel appelé **CEGAP** sous Matlab utilisant les méthodes décrites dans ce document.

Ce logiciel est opérationnel et des scéances de formation ont permis de présenter le logiciel à divers personnels du CEG : deux illustrations sont présentées dans la figure VI.32.

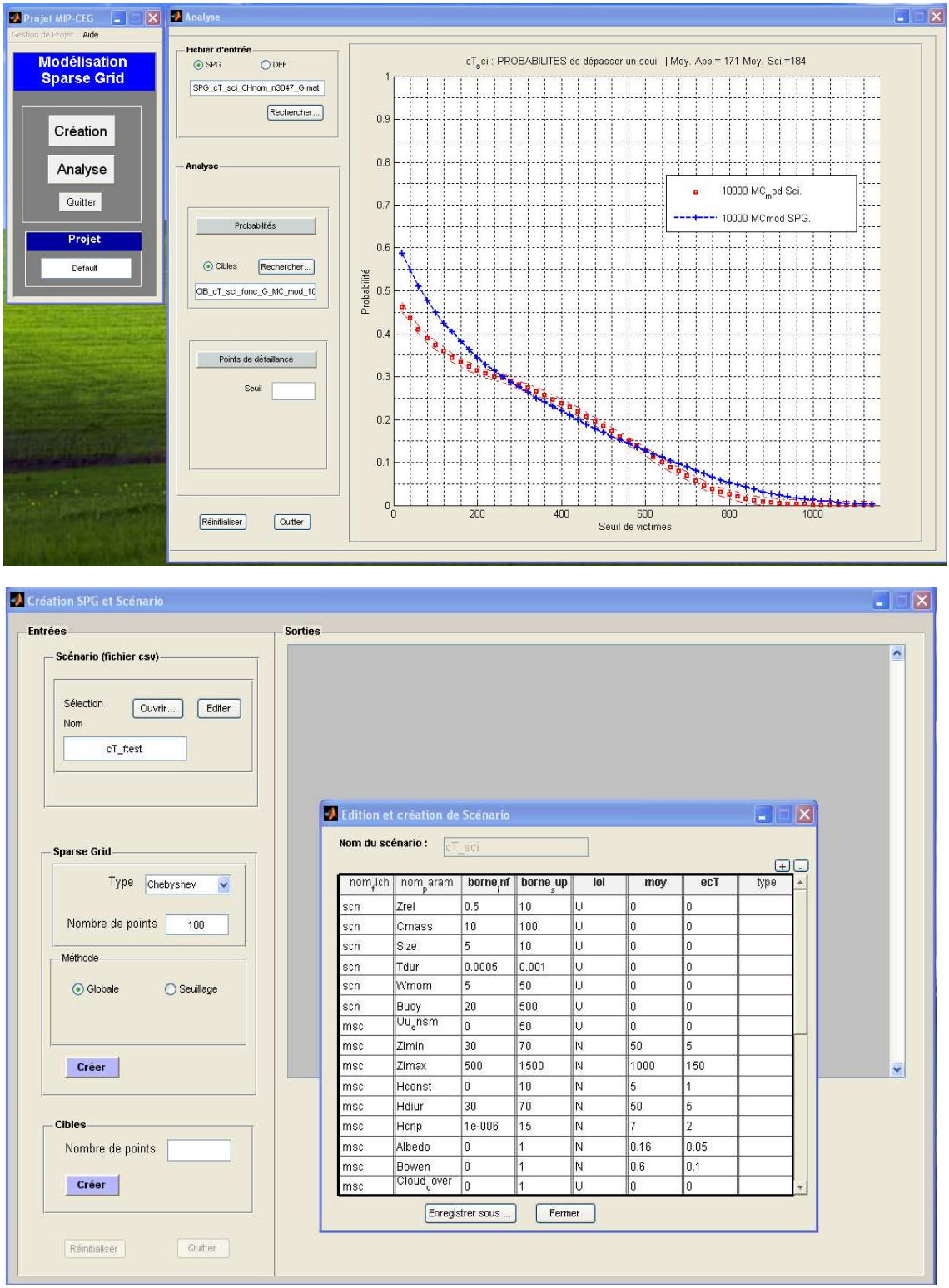


FIG. VI.32 – Captures d'écran du logiciel livré au CEG

Objectifs

L'objectif est à présent d'enrichir le code avec des recherches en cours au laboratoire MIP dans le cadre de la poursuite de la collaboration avec le CEG, de manière à

améliorer la méthode d'étude de fiabilité par approximation *sparse grid*.

Le logiciel CEGAP sera intégré aux logiciels de simulation d'accident chimique utilisés actuellement au CEG. Ils forment une chaîne de logiciels :

- le premier module constitue une modélisation 3D d'un site physique comportant un terrain naturel, des lieux d'habitation, et un site industriel contenant le produit toxique,
- le deuxième module simule l'explosion du conteneur de produit toxique et la génération du "terme source", qui décrit l'état du produit juste avant la dispersion,
- le troisième module est le logiciel SCIPUFF (utilisé dans notre étude) qui calcule la dispersion.

Les sorties des deux premiers modules deviennent ensuite la partie des entrées du logiciel SCIPUFF qui décrit l'état du produit et les paramètres de l'explosion, l'autre partie étant liée aux conditions naturelles (météorologiques,...).

La prochaine phase va consister à faire varier les nouveaux paramètres incertains liés aux deux premiers modules et effectuer l'étude de fiabilité sur des sorties intermédiaires (masse de produit éjecté, vitesse,...) et des sorties finales (à seuil de produit donné, surface «létale », rayon minimum et maximum de cette surface).

A plus long terme, la méthode d'approximation pourrait s'appliquer à d'autres domaines d'études du CEG utilisant des codes de simulation.

Perspectives

Il reste à confirmer les résultats de la méthode en dimension encore supérieure.

L'interpolation *sparse grid* possède de très bonnes propriétés pour traiter les problèmes en grande dimension, grâce à son approche de type hiérarchique et son mode adaptatif.

C'est une technique émergente, qui n'est pas encore utilisée dans le cadre des études de fiabilité, et qui présente des perspectives prometteuses.

Régularisation

Nous avons constaté cependant des problèmes de convergence dans le cas de la méthode globale sans déplacement des points de grille : quand la taille de la grille augmente, la précision de l'approximation se détériore.

Ce phénomène est connu dans le domaine de l'interpolation : des oscillations apparaissent quand on augmente le nombre de points, et la qualité de prédiction se dégrade car certains coefficients affectés aux fonctions de base deviennent trop grands.

Un **premier axe de recherche** consiste donc à appliquer une **technique de régularisation** lors de la création de la *sparse grid* g.

Si on note $B = (b_1, \ldots, b_n) \in \mathbb{R}^n$ les coefficients et $\{\phi_1, \ldots, \phi_n\}$ les fonctions de base alors $g = \sum_{j=1}^{n} b_j \ \phi_j$. On peut ajuster les coefficients calculés à un niveau de

grille donné, en appliquant une méthode de moindres carrés régularisés, c'est à dire en minimisant par rapport à B à la fois :

- la semi-norme H_1 de g, qui mesure ses variations, pour rendre plus petits ceux qui ont tendance à trop augmenter,
- l'écart au sens des moindres carrés entre l'approximation et la fonction de modélisation sur les points d'interpolation (cet écart vaut zéro avant régularisation).

Démarche adaptative

Nous avons constaté aussi l'importance en grande dimension de disposer une démarche adaptative : à cause de la limitation du coût de calcul, on ne peut espérer obtenir une approximation globalement précise.

Il faut construire l'approximation en disposant les données d'évaluations, c'est à dire l'information, dans les zones importantes pour le calcul :

- le **mode adaptatif** des *sparse grids* qui concentre le coût de calcul dans les **directions importantes** détectées au fur et à mesure adopte ce point de vue,
- la technique du **seuillage** filtre les informations pour ne conserver qu'une version simplifiée (au-dessus ou au-dessous du seuil) mais suffisante pour le but recherché du **calcul de probabilité**,
- l'approche par **approximations successives** utilise l'information apportée sur le **point de conception** par une première approximation afin de converger vers celui-ci.

Dans le cas de la méthode des réseaux de neurones, l'avantage était de pouvoir disposer les données d'apprentissage selon un tirage respectant les lois de probabilité du scénario, mais cette possibilité n'existe pas avec la grille fixe des *sparse grids*.

Le déplacement des points de grille vers le point nominal améliore l'approche globale, mais présente l'inconvénient de pas être valable au-delà du domaine de taille $3\ \sigma$ dans chaque direction : au-delà la *sparse grid* n'est pas définie et une partie du domaine est absente de l'approximation.

Un **deuxième axe de recherche** serait par exemple d'effectuer un changement de variable sur la fonction de modélisation pour transformer les lois gaussiennes en lois uniformes afin que l'**apprentissage de la *sparse grid* s'effectue sur des données mieux réparties autour du point nominal** sans la restriction précédente.

Décomposition de la variance

Etant donné que les **polynômes de Chebyshev** utilisés dans les ***sparse grids*** sont **orthogonaux**, un **troisième axe de recherche** consisterait à adapter la méthode de décomposition de la variance basée sur le **développement en polynômes de chaos** (voir sections 3.3 page 162 et 5.3.2 page 175), afin d'enrichir les **mesures de sensibilité** par le calcul d'indices de type Sobol.

Conclusion générale

Dans cette thèse, nous avons participé à deux projets de recherche, abordant **divers aspects des problèmes d'optimisation**.

Problème de conception d'une antenne spatiale

Dans la collaboration avec Thalès Alenia Space, nous avons **modélisé une antenne spatiale réseau active**, et notre contribution a porté sur les points suivants :

- **Synthèse du réseau**

  En **reformulant de manière adaptée le problème d'optimisation afin de le simplifier**, nous avons construit une méthode efficace pour calculer les lois d'alimentation avec un algorithme de descente.

- **Optimisation topologique**

  En utilisant une **décomposition en valeurs singulières des lois optimales** couplé à un **algorithme d'optimisation géométrique de type gradient topologique**, nous avons regroupé les sources élémentaires et obtenu une antenne satisfaisant les contraintes industrielles spécifiées.

Cette approche originale a donné de bons résultats, et a permis de mettre au point un **logiciel d'aide à la conception** de ce type d'antennes.

Analyse des conséquences de la dispersion d'un produit

Dans la collaboration avec le Centre d'Etudes de Gramat, nous avons utilisé un code de calcul de dispersion d'un produit dans l'environnement.

Nous avons **modélisé un scénario d'accident de type chimique**, défini par un grand nombre de **paramètres incertains**, et notre contribution a porté sur les points suivants :

- **Etude de fiabilité et de sensibilité**

  En contruisant un **modèle approché de la fonction de modélisation**, nous avons pu utiliser les **gradients de l'approximation** afin de calculer un point de conception et déterminer les paramètres influents.

- **Méthodes d'approximation**

  Nous avons comparé une approximation par **réseau de neurones** et une interpolation sur grille éparse (***sparse grid***).

  Les *sparse grids* ont permis d'obtenir de meilleurs résultats grâce à leur caractère hiérarchique et à un algorithme adaptatif.

- **Grande dimension**

  En **grande dimension**, avec le cas d'une fonction difficile à approcher globalement, nous avons utilisé les *sparse grids*

  - avec un algorithme d'**approximations successives** qui permet de converger vers le point de conception,
  - avec un algorithme de **seuillage des données** qui permet de diminuer le coût de calcul des probabilités.

Cette méthode a été implémentée dans un **logiciel d'aide à la décision** qui effectue une étude de fiabilité (probabilités de défaillance et point de conception) et de sensibilité (paramètres influents) pour un coût de calcul raisonnable.

Bibliographie

[Ach03] S. Achatz. Higher order sparse grid methods for elliptic partial differential equations with variable coefficients. *Computing*, Vol. 71(1) :p. 1–15, 2003.

[ADGJT05] G. Allaire, F. De Gournay, F. Jouve, and A.M. Toader. Structural optimization using topological and shape sensitivity via a level set method. *Control and Cybernetics*, Vol. 34 :p. 59–80, 2005.

[All01] G. Allaire. *Shape optimization by the homogenization method*. Springer, 2001.

[ASS01] A. C. Antoulas, D. C. Sorensen, and Gugercin S. A survey of model reduction methods for large-scale systems. *Contemporary Mathematics*, Vol. 280 :p. 193–219, 2001.

[Bal97] C. A. Balanis. *Antenna Theory : Analysis and Design*. Wiley, 1997.

[Bar93] A. Barron. Universal approximation bounds for superposition of a sigmoidal function. *IEEE Transactions of Information Theory*, Vol. 39 :p. 930–945, 1993.

[BCC+92] C. Bischof, A. Carle, G. Corliss, A. Griewank, and P. Hovland. Adifor generating derivative codes from fortran programs. Technical report, Mathematics and Computer Science Division, Argonne National Laboratory, 1992.

[BDF76] F.B. Bretherton, R.E. Davis, and C.B. Fandry. A technique for objective analysis and design of oceanographic experiments applied to mode-73. *Deep-Sea Res.*, Vol. 23 :p. 559–582, 1976.

[BEMP94] O.M. Bucci, G. Elia, G. Mazzarella, and G. Panariello. Antenna pattern synthesis : a new general approach. *Proceedings of the IEEE*, Vol. 82, Issue 3 :p. 358–371, 1994.

[Ben89] M.P. Bendsoe. Optimal shape design as a material distribution problem. *Structural optimization*, Vol. 1 :p. 193–202, 1989.

[BFT97] O.M. Bucci, N. Fiorentino, and Isernia T. An efficient approach to 2-d arrays mask constrained power pattern synthesis. *Antennas and Propagation Society International Symposium, IEEE*, Vol. 4 :p. 2252–2255, 1997.

[BG04] H.J. Bungartz and M. Griebel. Sparse grids. *Acta Numerica*, Vol. 13 :p. 147–269, 2004.

[BGLS03] J.F. Bonnans, J.C. Gilbert, C. Lemaréchal, and C. Sagastizabal. *Numerical optimization*. Springer-Verlag, 2003.

[Bje88] P. Bjerager. Probability integration by directional simulation. *Journal of Engineering Mechanics*, Vol. 114(n°8) :p. 1285–1302, 1988.

[BK88] M.P. Bendsoe and N. Kikuchi. Generating optimal topologies in structural design using a homogenization method. *Computer methods in applied mechanics and engineering*, Vol. 71 :p. 197–224, 1988.

[BNR00] V. Barthelmann, E. Novak, and K. Ritter. High dimensional polynomial interpolation on sparse grids. *Adv. Comput. Math.*, Vol. 12(4) :p. 273–288, 2000.

[Bou04] J. Boutahar. *Méthodes de réduction et de propagation d'incertitudes : application à un modèle de Chimie-Transport pour la modélisation et la simulation des impacts*. PhD thesis, Ecole Nationale des Ponts et Chaussées, 2004.

[Bre84] K. Breitung. Asymptotic approximations for multinormal integrals. *Journal of Engineering Mechanics Division ASCE*, Vol. 110(3) :p. 357–366, 1984.

[BS92] P.L. Butzer and R.L. Stens. Sampling theory for not-necessarily band-limited functions : A historical overview. *SIAM Review*, Vol. 34 :p. 40–53, 1992.

[BS03] M.P. Bendsoe and O. Sigmund. *Topology optimizationn : theory, methods and applications*. Springer, 2003.

[Cal04] S. Calvel. *Conception d'organes automobiles par optimisation topologique*. PhD thesis, Université Toulouse III, 2004.

[Cap06] Elia G. Capozzoli. Global optimization and antennas synthesis and diagnosis, part one : concepts, tools, stategies and performances. *PIER (Progress In Electromagnetics Research)*, Vol. 56, 2006.

[CCG+07] G. Caille, Y. Cailloce, C. Guiraud, D. Auroux, T. Touya, and M. Masmousdi. Large multibeam array antennas with reduced number of active chains. *Proceedings of the 2d Euopean Conference on Antennas and Propagation*, 2007.

[CGGM00] J. Cea, S. Garreau, P. Guillaume, and M. Masmousdi. The shape and topological optimizations connection. *Computational methods in applied mechanics and engineering*, Vol. 188 :p. 713–726, 2000.

[Che71] D. Cheng. Optimization techniques for antenna arrays. *Proceedings of the IEEE*, Vol. 59, n°12 :p. 1664–1674, 1971.

[Cia98] P. G. Ciarlet. *Introduction à l'analyse numérique matricielle et à l'optimisation*. Masson, 1998.

[CLG00] R. Caruana, S. Lawrence, and L. Giles. Overfitting in neural nets : Back propagation, conjugated gradient, and early stopping. *Neural Information Processing Systems*, 2000.

[CLS78] R.I. Cukier, H.B. Levine, and K.E. Schuler. Nonlinear sensitivity analysis of multiparameter model systems. *Journal Comput. Phys.*, Vol. 26 :p. 1–42, 1978.

[DA96] E. Dubois and J. Audet. Global optimization of arrays by simulated annealing and genetic algorithms,. In *Conference on Underwater Defense Technology*, London, juillet 1996.

[DA97] E. Dubois and J. Audet. Global time-domain optimization of multilayees by simulated annealing and genetic algorithms. In *Proc. of the second world congress 97 of struct. and multi. optimi.*, pages p. 55–60, 1997.

[DFG97] J.J. Droesbeke, J. Fine, and Saporta G. *Plans d'expériences : Application à l'entreprise*. Technip, 1997.

[DHV92] E. Damslet, A. Hage, and R. Volden. Maximum information at minimum cost. a north field development study with an experimental design. *JPT*, page 1350, December 1992.

[DJ06] Saporta G. Droesbeke J.J., Lejeune M. *Analyse statistique des données spatiales*. Technip, 2006.

[DKD98] A. Der Kiureghian and T. Dakessian. Multiple design points in first and second-order reliability. *Structural Safety*, Vol. 20(1) :p. 37–50, 1998.

[DKP86] A. Der Kiureghian and Liu P.L. Structural reliability under incomplete probability information. *English Mechanics Division ASCE*, Vol. 112(1) :p. 85–104, 1986.

[DM96] O. Ditlevsen and H.O. Madsen. *Structural Reliability Methods*. Wiley, 1996.

[DMS+02] G. Dreyfus, J.M. Martinez, M. Samuelides, M.B. Gordon, F. Badran, S. Thiria, and L. Herault. *Réseaux de neurones : Méthodologie et applications*. Eyrolles, 2002.

[Dol46] Dolph. A current distribution for broadside arrays which optimizes the relationship between beam width and side lobe level. *Proceedings of the IRE*, Vol. 34 :p. 335–348, 1946.

[DPST03] Dreo, Petrowski, Siarry, and Taillard. *Métaheuristiques pour l'optimisation difficile*. Eyrolles, 2003.

[Dra86] A. Drabowitch. *Antennes, Tome 2 (Applications)*. Masson, 1986.

[Dub97] E. Dubois. *Méthodes d'optimisation globale appliquée à l'acoustique sous marine*. PhD thesis, Université de Nice Sophia Antipolis, 1997.

[EKS94] H.A. Eschenauer, V.V. Kobolev, and A. Schumacher. Bubble method for topology and shape optimization of structures. *Structural optimization*, Vol. 8 :p. 42–51, 1994.

[Fad05] N. Fadlallah. *Contribution à l'optimisation de la synthèse du lobe de rayonnement pour une antenne intelligente. Application à la conception de réseaux à déphasage.* PhD thesis, Université de Limoges, 2005.

[FHP96] K. Frank, S. Heinrich, and S. Pereverzev. Information complexity of multivariate fredholm integral equations in sobolev classes. *Journal of Complexity*, Vol. 12 :p. 17–34, 1996.

[Fis96] G.S. Fishman. *Monte-Carlo : Concepts, Algorithms and Applications.* Springer-Verlag, 1996.

[FP98] C. Faure and Y. Papegay. Odyssee user's guide (version 1.7). Technical report, INRIA, 1998.

[Gan65] I.S. Gandin. Objective analysis of meteorological fields. *Israël Program for Scientific Translations*, Vol. 1373 :p. 242, 1965.

[Gar83] C.W. Gardiner. *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences.* Springer-Verlag, 1983.

[GG03] T. Gerstner and M. Griebel. Dimension-adaptative tensor-product quadrature. *Computing*, Vol. 71 :p. 65–87, 2003.

[GGM01] S. Garreau, P. Guillaume, and M. Masmoudi. The topological asymptotic for pde systems : the elasticity case. *SIAM Journal Control Optimal*, Vol. 39 :p. 1756–1778, 2001.

[GK00] M Griebel and S Knapek. Optimized tensor-product approximation spaces. *Constructive Approximation*, Vol. 16(4) :p. 525–540, 2000.

[Gou96] J. Goupy. *La méthode des plans d'expériences : optimisation du choix des essais et de l'interprétation des résultats.* Dunod, 1996.

[Gou99] J. Goupy. *Plans d'expériences pour surfaces de réponse.* Dunod, 1999.

[GP91] R.G. Ghanem and Spanos P.D. *Stochastic Finite Elements : a spectral approach.* Springer, 1991.

[Gri91] M. Griebel. Parallel algorithms for partial diferential equations. *Notes on Numerical Fluid Mechanics*, Vol. 31 :p. 94–100, 1991.

[Gri00] A. Griewank. Evaluating derivatives : principles and techniques of algorithmic differentiation. *Frontiers in Applied Mathematics, Society of Industrial and Applied Mathematics (SIAM)*, Vol. 19, 2000.

[Gui07] C. Guiraud. Reducing dra complexity by thinning and splitting into non-regular subarrays. In *29th ESA Antenna Workshop*, April 2007.

[GVL96] G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins University Press (Baltimore), 1996.

[Hay99] S. Haykin. *Neural Networks, a comprehensive foundation.* Society for Industrial and Applied Mathematics (SIAM), 1999.

[HHL93] J.B. Hiriart-Hurruty and C. Lemarechal. *Convex analysis and minimization algorithms Vol.1 & 2.* Springer, 1993.

[HL74] A.M. Hasofer and N.C. Lind. Exact end invariant second moment code format. *Journal Eng. Mechanics Div.*, Vol. 100 :p. 111–121, 1974.

[HM00] A. Haldar and S. Mahadevan. *Probability, Reliability and Statistical Methods in Engineering Design.* Wiley, 2000.

[Hor89] K. Hornik. Multilayer feedforward networks are universal approximators. *Neural Networks*, Vol. 2 :p. 359–366, 1989.

[HT02] L. Harry and Van Trees. *Optimum Array Processing Part IV.* Wiley, 2002.

[IH90] R.L. Iman and S.C. Hora. A robust measure of uncertainty importance for use in fault tree system analysis. *Risk Analysis*, Vol. 10 :p. 401–406, 1990.

[Isu99] S.S. Isukapalli. *Uncertainty analysis of transport-transformation models.* PhD thesis, Rutgers State University of New Jersey, 1999.

[JCS01] R. Jin, W. Chen, and T. W. Simpson. Comparative studies of metamodeling techniques under multiple modeling criteria. *Journal of Structural and Multidisciplinary Optimization*, Vol. 23(1) :p. 1–13, 2001.

[JM90] M. E. Johnson and D. Moore, L. M.and Ylvisaker. Minimax and maximin distance designs. *Journal of Statistical Planning and Inference*, Vol. 26 :p. 131–148, 1990.

[Kel99] C.T. Kelley. Iterative methods for optimization. *Frontiers in Applied Mathematics, Society of Industrial and Applied Mathematics (SIAM)*, Vol. 18, 1999.

[Kir96] A. Kirsch. *An introduction to the mathematical theory of inverse problems.* Springer Verlag, 1996.

[Kli05] W.A. Klimke. *Uncertainty Modeling using Fuzzy Arithmetic and Sparse Grids.* PhD thesis, Stuttgart University, 2005.

[KM97] A. Kumar and P. Murthy. Synthesis of equally excited linear arrays. *IEEE Transactions on Antennas and Propagation*, Vol. 25, Issue 3 :p. 425–428, 1997.

[Lam93] R. Lamberti. *Synthèse de réseau d'antennes et méthodes adaptatives avec contraintes de diagramme en ondes ionosphériques décamétriques.* PhD thesis, Université Paris 11, 1993.

[Lem05] M. Lemaire. *Fiabilité des structures : couplage mécano-fiabiliste statique.* Hermes science, 2005.

[Lev44] K. Levenberg. A method for the solution of certain problems in least squares. *Quarterly of Applied Mathematics*, Vol. 2 :p. 164–168, 1944.

[LPS98] B. Lapeyre, E Pardoux, and R. Sentis. *Méthodes de Monte-Carlo pour les équations de transport et de diffusion.* Springer, 1998.

[Man89] C. Mangenot. *Méthode de synthèse de réseaux linéaires et plans rayonnant un diagramme à contour formé.* PhD thesis, Université Toulouse III, 1989.

[Mar63] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal of Applied Mathematics*, Vol. 11 :p. 431–441, 1963.

[Mas87] M. Masmoudi. Outils pour l'optimisation de forme. Thèse d'etat, Université de Nice, 1987.

[Mas01] M. Masmoudi. The topological asymptotic. *Computational methods for control applications*, Vol. 16 :p. 53–72, 2001.

[Mat63] G. Matheron. Principles of geostatistics. *Economic Geology*, Vol. 58 :p. 1246–1266, 1963.

[MBC79] M. D. McKay, R. J. Beckman, and W. J. Connover. A comparison of three methods for selecting values of input variables in the analysis of output from computer code. *Technometrics*, Vol. 21 :p. 239–245, 1979.

[Min83] M. Minoux. *Programmation mathématique : théorie et algorithmes.* Dunod, 1983.

[MKL86] H.O. Madsen, S. Krenk, and N.C. Lind. *Methods of structural safety.* Prentice Hall Inc., 1986.

[MM95] R.H. Myers and D.C. Montgomery. *Response Surface Methodology : Process and Product Optimization Using Designed Experiments.* Wiley, New York, 1995.

[Moo79] R.E Moore. Methods and applications of interval analysis. *Society of Industrial and Applied Mathematics (SIAM), Philadelphia*, 1979.

[Mor91] M.D. Morris. Factorial sampling plans for preliminary computational experiments. *Technometrics*, Vol. 33 :p. 161–174, 1991.

[MP93] G. Mazzarella and G. Panariello. Pattern synthesis of conformal arrays. *Antennas and Propagation Society International Symposium*, Vol. 2 :p. 1054–1057, 1993.

[Nat62] A. Nataf. Détermination des distributions dont les marges sont données. *Comptes Rendus de l'Académie des Sciences*, Vol. 225 :p. 42–43, 1962.

[NV95] S.R. Nagesh and T.S. Vedavathy. A procedure for synthesizing a specified sidelobe topography using an arbitrary array. *IEEE Transactions on Antennas and Propagation*, Vol. 43, Issue 7 :p. 742–745, 1995.

[NW99] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, 1999.

[PM85] E. Polak and D.Q. Mayne. Algorithm models for nondifferentiable optimization. *SIAM Journal on Control and Optimization*, Vol. 23, Issue 3 :p. 477–491, 1985.

[Rac79] B. Rackwitz, R. ansd Fiessler. Structural reliability under combined random load sequences. *Computers and Structures*, Vol. 9 :p. 489–494, 1979.

[RASS99] H. Rabitz, O. Alis, J. Shorter, and K. Shim. Efficient input-output model representations. *Comput. Phys. Commun.*, Vol. 117 :p. 11–20, 1999.

[Roq98] C. Roques. *Optimisation des performances minimales garanties d'une antenne réseau en présence d'erreurs aléatoires et de pannes.* PhD thesis, Université Toulouse III, 1998.

[Ros52] M. Rosenblatt. Remarks on a multivariate transformation. *The Annals of Mathematical Statistics*, Vol. 23 :p. 470–472, 1952.

[Sam04] B. Samet. *L'analyse asymptotique topologique pour les équations de Maxwell et applications.* PhD thesis, Université Toulouse III, 2004.

[Sap06] G. Saporta. *Probabilités, analyse des données et statistique.* Technip, 2006.

[SBG+02] T.W. Simpson, A.J. Booker, D. Ghosh, Giunta A., P. Koch, and R.J. Yang. Approximation methods in multidisciplinary analysis and optimization : A panel discussion. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis & Optimization*, Atlanta, GA, September 2-4 2002.

[Sch01] G.I. Schueller. Computational stochastics mechanics - recent advances. *Computers and Structures*, Vol. 79 :p. 2225–2234, 2001.

[Sch06] C. Scheidt. *Analyse statistique d'expériences simulées : Modélisation adaptative de réponses non-régulières par krigeage et plans d'expériences.* PhD thesis, Université Louis Pasteur Strasbourg I, 2006.

[SCS00] A. Saltelli, K. Chan, and E.M. Scott. *Sensitivity analysis.* Wiley, 2000.

[SJM76] H. Schjaer-Jacobsen and K. Madsen. Synthesis of nonuniformly spaced arrays using a general nonlinear minimax optimisation method. *IEEE Transactions on Antennas and Propagation*, Vol. 24, Issue 4 :p. 501–506, 1976.

[SLC01] T.W. Simpson, D.K.J. Lin, and W. Chen. Sampling strategies for computer experiments : Design and analysis. *International Journal of Reliability and Applications*, Vol. 2(3) :p. 209–240, 2001.

[Smo63] S.A. Smolyak. Quadrature and interpolation formulas for tensor products of certain classes of functions. *Soviet Math. Dokl.*, Vol. 4 :p. 240–243, 1963.

[Sob93] I.M. Sobol. Sensitivity estimates for non linear mathematical models. *Mathematical Modeling Comput. Experiment*, Vol. 1 :p. 407–414, 1993.

[SPH+98] R. I. Sykes, S. F. Parker, D. S. Henn, C. P. Cerasoli, and L. P. Santos. Pc-scipuff version 1.1pd.1 technical documentation. ARAP Report No. 718, Titan Research & Technology Division, ARAP Group, P.O. Box 2229, Princeton, NJ, 08543-2229, 1998.

[Spr98] F. Sprengel. Periodic interpolation and wavelets on sparse grids. *Numerical Algorithms*, Vol. 17(1-2) :p. 147–169, 1998.

[STCM04] A. Saltelli, S. Tarantola, F. Campolongo, and Ratto M. *Sensitivity analysis in practice - a guide to assessing scientific models*. Wiley, 2004.

[SWMW89] J. Sacks, W. Welch, T. Mitchell, and P. Wynn. Design and analysis of computer experiments (with discussion). *Statistical Science*, Vol. 4 :p. 409–435, 1989.

[SZ92] J. Sokolowski and J.P. Zolesio. Introduction to shape optimization : shape sensitivity analysis. *Springer series in computational mathematics*, Vol. 10, 1992.

[SZ99] J. Sokolowski and A. Zochowski. On the topological derivative in shape optimization. *SIAM Journal Control Optimal*, Vol. 37 :p. 1251–1272, 1999.

[TA08] T. Touya and D. Auroux. Control and topological optimization of a large multibeam array antenna. *Proceedings of Antennas Radar and Wave Propagation*, 2008.

[Tat95] M.A. Tatang. *Direct Incorporation of Uncertainty in Chemical and Environmental Engineering Systems*. PhD thesis, MIT, 1995.

[Tho90] Thourel. *Les antennes, Tome 2 (Calcul et conception des dispositifs en ondes centimétriques et millimétriques)*. Cepadues, 1990.

[VG04] M. Van Grieken. *Optimisation pour l'apprentissage et apprentissage pour l'optimisation*. PhD thesis, Université Toulouse III, 2004.

[WEF+01] S. Wojtkiewicz, M. Eldred, R. Field, J. Urbina, and J. Red-Horse. Uncertainty quantification in large computational engineering models, 2001.

[Yar03] A.R. Yarahmadi. *La méthode des groupes-clef probabiliste*. PhD thesis, Ecole des Mines de Nancy, 2003.

[Zen91] C. Zenger. Sparse grids : parallel algorithms for partial differential equations. *Proceedings of the Sixth GAMM-Seminar*, Vol. 31 :p. 241–251, 1991.

[ZO99] Y.G. Zhao and T. A Ono. A general procedure for first/second-order reliability method (form/sorm). *Structural Safety*, Vol. 21 :p. 95–112, 1999.

Annexe A

Calculs de la première partie

1 Minimiser le maximum est convexe

1.1 Lemme

– Soient $g_1, g_2 : \mathbb{R}^N \longrightarrow \mathbb{R}$, et $\lambda_1,\ \lambda_2 \geqslant 0$

alors

$$\max_X(\lambda_1 g_1(X) + \lambda_2 g_2(X)) \leqslant \lambda_1 \max_X(g_1(X)) + \lambda_2 \max_X(g_2(X))$$

1.2 Proposition

– Soient $f : \mathbb{R}^{2n} \times \mathbb{R}^2 \longrightarrow \mathbb{R}$ une fonction convexe par rapport à a

$$(a\ ,\ x) \longmapsto f(a,x)$$

et Ω un domaine de $\mathbb{R}^2$,

alors $g : \mathbb{R}^{2n} \longrightarrow \mathbb{R}$ est convexe.

$$a \longmapsto g(a) = \max_{x\in\Omega} f(a,x)$$

– **Conséquence** :

Appliqué à la directivité $\mathcal{D}(a,x)$, si on note Ω_0 l'intérieur du spot, on obtient :

$$a \longmapsto \min_a \left(\max_{x\in\Omega\setminus\Omega_0} \mathcal{D}(a,x) \right) \text{ est convexe}$$

donc admet un unique minimum a_0.

– **Preuve** :

$$f \text{ convexe en } a$$

$\Leftrightarrow \forall a, b \in \mathbb{R}^{2n} \quad \forall \alpha \in [0,1] \quad \forall x \in \Omega$

$$f(\alpha a + (1-\alpha)b, x) \leqslant \alpha f(a,x) + (1-\alpha) f(b,x)$$

$$\Leftrightarrow \forall a, b \in \mathbb{R}^{2n} \quad \forall \alpha \in [0,1] \quad \forall x \in \Omega$$

$$f(\alpha a + (1-\alpha)b, x) \leqslant \max_{x \in \Omega}(\alpha f(a,x) + (1-\alpha) f(b,x))$$

$$\Leftrightarrow \forall a, b \in \mathbb{R}^{2n} \quad \forall \alpha \in [0,1] \quad \forall x \in \Omega \qquad \text{on utilise le lemme}$$

$$f(\alpha a + (1-\alpha)b, x) \leqslant \alpha \max_{x \in \Omega}(f(a,x)) + (1-\alpha) \max_{x \in \Omega}(f(b,x))$$

$$\Rightarrow \forall a, b \in \mathbb{R}^{2n} \quad \forall \alpha \in [0,1]$$

$$\max_{x \in \Omega}(f(\alpha a + (1-\alpha)b, x)) \leqslant \alpha \max_{x \in \Omega}(f(a,x)) + (1-\alpha) \max_{x \in \Omega}(f(b,x))$$

$$\Rightarrow \forall a, b \in \mathbb{R}^{2n} \quad \forall \alpha \in [0,1]$$

$$g(\alpha a + (1-\alpha)b) \leqslant \alpha g(a) + (1-\alpha) g(b)$$

$$\Rightarrow$$

$$g \text{ convexe en } a$$

2 Calcul de la directivité de l'antenne $\widetilde{\mathcal{D}}(a, x)$

2.1 Calcul du diagramme de rayonnement de l'antenne $E(a, x)$

2.1.1 Données

- Soit une antenne constituée de n ER rectangulaires disposés en réseau plan.
- Chaque ER rectangulaire de centre M_j par rapport au centre de l'antenne est constitué de Q_j patches en hauteur et P_j patches en longueur.

  On a donc $M_j \times Q_j = n_j$ patches pour cet ER.

  On note N_j le centre d'un patch p de l'ER j.
- λ est la longueur d'onde.
- d est la distance entre deux patches.
- L'alimentation est uniforme sur les patches du même ER, et le rayonnement d'une source ou patch est $\tilde{e}(x)$.

2.1.2 Calcul

On applique le théorème de superposition à l'ensemble des sources élémentaires de l'antenne, qui sont les patches, pour calculer le diagramme de rayonnement :

$$E(a, x) = \sum_{j=1}^{n \times n_j} a_j \; e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{ON_j}, \overrightarrow{Ox}\rangle} \tilde{e}_x$$

On regroupe les patches par ER :

$$E(a, x) = \sum_{j=1}^{n} \sum_{J=1}^{n_j} a_{\mathbb{C}_{J_j}} e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{ON_{J_j}}, \overrightarrow{Ox}\rangle} \; \tilde{e}_x$$

$$E(a, x) = \sum_{j=1}^{n} a_j (\underbrace{\sum_{J=1}^{n_j} e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{ON_{J_j}}, \overrightarrow{Ox}\rangle}_{=E_1}}) \; \tilde{e}_x$$

On calcule la contribution d'un ER :

$$E_1 = \sum_{J=1}^{n_j} e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{OM_J} + \overrightarrow{M_J N_J}, \overrightarrow{Ox}\rangle}$$

$$E_1 = e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{OM_J}, \overrightarrow{Ox}\rangle} \underbrace{\sum_{J=1}^{n_J} e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{O\widetilde{N}_J}, \overrightarrow{Ox}\rangle}}_{=E_2}$$

avec $\widetilde{N}_J$ le centre d'un patch p par rapport cette fois au centre de l'ER j, on a $\overrightarrow{M_J N_J} = \overrightarrow{O\widetilde{N}_J}$.

On renumérote les $\widetilde{N}_J$ en $\widetilde{N}_{J_1J_2}$ car $n_j = Q_jP_j$.

$$E_2 = \sum_{J_1=1}^{Q_j}\sum_{J_2=1}^{P_j} e^{i\frac{2\pi}{\lambda}\langle\overrightarrow{O\widetilde{N}_{J_1J_2}},\overrightarrow{Ox}\rangle}$$

$$E_2 = \sum_{J_1=1}^{Q_j}\sum_{J_2=1}^{P_j} e^{i\frac{2\pi}{\lambda}\langle\overrightarrow{O\widetilde{N}_{1,1}}+\overrightarrow{\widetilde{N}_1\widetilde{N}_{J_1J_2}},\overrightarrow{Ox}\rangle}$$

$$E_2 = e^{i\frac{2\pi}{\lambda}\langle\overrightarrow{O\widetilde{N}_{1,1}},\overrightarrow{Ox}\rangle}\sum_{J_1=1}^{Q_j}\sum_{J_2=1}^{P_j} e^{i\frac{2\pi}{\lambda}\langle\overrightarrow{\widetilde{N}_1\widetilde{N}_{J_1J_2}},\overrightarrow{Ox}\rangle}$$

On passe aux coordonnées des vecteurs et on pose

$$\overrightarrow{\widetilde{N}_1\widetilde{N}_{J_1J_2}} = \begin{pmatrix}\tilde{x}_{J_2}\\ \tilde{y}_{J_1}\end{pmatrix} = \begin{pmatrix}(J_2-1)d\\ -(J_1-1)d\end{pmatrix}$$

et

$$\overrightarrow{O\widetilde{N}_{1,1}} = \begin{pmatrix}x_{\tilde{N}_1}\\ y_{\tilde{N}_1}\end{pmatrix} = \begin{pmatrix}-\frac{P_j}{2}d+\frac{d}{2}\\ \frac{Q_j}{2}d-\frac{d}{2}\end{pmatrix} = \begin{pmatrix}(-P_J+1)\frac{d}{2}\\ (\ Q_J-1)\frac{d}{2}\end{pmatrix}$$

$$E_2 = e^{i\frac{2\pi}{\lambda}(\tilde{x}_1u+\tilde{y}_1v)}\sum_{J_1=1}^{Q_j}\sum_{J_2=1}^{P_j} e^{i\frac{2\pi}{\lambda}(\tilde{x}_{J_2}u+\tilde{y}_{J_1}v)}$$

$$E_2 = e^{i\frac{2\pi}{\lambda}\tilde{y}_1v}\sum_{J_1=1}^{Q_j} e^{i\frac{2\pi}{\lambda}\tilde{y}_{J_1}v}\underbrace{(e^{i\frac{2\pi}{\lambda}\tilde{x}_1u}\sum_{J_2=1}^{P_j} e^{i\frac{2\pi}{\lambda}\tilde{x}_{J_2}u})}_{=E_{\tilde{x}}}$$

$$E_{\tilde{x}} = e^{i\frac{2\pi}{\lambda}(-(P_J+1)\frac{d}{2}u)}\sum_{J_2=1}^{P_j} e^{i\frac{2\pi}{\lambda}(J_2-1)du}$$

On pose $J_2' = J_2 - 1$ et $\psi_u = \frac{\pi du}{\lambda}$

$$E_{\tilde{x}} = e^{i\psi_u(-P_J+1)}\sum_{J_2'=0}^{P_j-1}(e^{i2\psi_u})^{J_2'}$$

On obtient une série géométrique de raison $r = i2\psi_u$ et la simplification fondamentale de la formule avec $\sum_{k=0}^{N} r^k = \frac{r^{N+1}-1}{r-1}$

$$E_{\tilde{x}} = e^{i\psi_u(-P_J+1)}\frac{e^{i2\psi_uP_j}-1}{e^{i2\psi_u}-1}$$

$$E_{\tilde{x}} = e^{i\psi_u(-P_J+1)}\frac{e^{i2\psi_uP_j}-1}{e^{i2\psi_u}-1}$$

$$E_{\tilde{x}} = \underbrace{e^{i\psi_u(-P_J+1)}\frac{e^{i\psi_u P_j}}{e^{i\psi_u}}}_{=e^{i0}=1}\ \frac{2i\ \sin(\psi_u P_J)}{2i\ \sin(\psi_u)}$$

$$E_{\tilde{x}} = \frac{\sin(\psi_u P_j)}{\sin(\psi_u)}$$

En faisant le même type de calcul, on trouve

$$E_{\tilde{y}} = \frac{\sin(\psi_v Q_j)}{\sin(\psi_v)}$$

Finalement, on trouve :

Diagramme de réseau de l'antenne :

$$E(a,x) = \left(\sum_{j=1}^{n}\ a_j\ e^{i\frac{2\pi}{\lambda}\langle\overrightarrow{OM_j},\overrightarrow{Ox}\rangle}\ \frac{\sin(\psi_u P_j)}{\sin(\psi_u)}\frac{\sin(\psi_v Q_j)}{\sin(\psi_v)}\right)\ \tilde{e}(x)$$

avec

$$\psi_u = \frac{\pi du}{\lambda} \quad \text{et} \quad \psi_v = \frac{\pi dv}{\lambda}$$

2.1.3 Calcul du champ rayonné par un patch

On utilise le passage des coordonnées sphériques aux coordonnées cartésiennes pour calculer θ :

$$\begin{pmatrix} \theta \\ \varphi \end{pmatrix} = \begin{pmatrix} \arctan\left(\frac{\sqrt{u^2+v^2}}{w}\right) \\ \arctan\left(\frac{v}{u}\right) \end{pmatrix} \qquad \text{avec } w = \sqrt{1-(u^2+v^2)}$$

2.1.4 Calcul d'une intégrale de surface

Soit le calcul d'une intégrale de surface de variable $x = \begin{pmatrix} u \\ v \end{pmatrix}$:

$$\tilde{\mathcal{I}} = \iint_{\Phi} f(x)\ dudv$$

Pour calculer la valeur numérique, il faut discrétiser l'intégrale sur les points x_k du maillage Terre contenus dans le domaine de définition Φ :

- On découpe le domaine Φ en zones élémentaires $d\Lambda_k$, $k = 1 \dots p$ de manière à ce que

$$\sum_{k=1}^{p} d\Lambda_k = \Phi$$

– Dans chaque zone élémentaire $d\Lambda_k$, on choisit un point x_k en lequel on évalue f.

– L'intégrale est transformée en somme discrète, et notée $\mathcal{I}$:

$$\mathcal{I} = \sum_{x_k \in \ \Phi} f(x_k)\ d\Lambda_k \longrightarrow_{k\to\infty} \tilde{\mathcal{I}}$$

2.1.5 Calcul de la puissance

Dans le cas de l'intégrale qui mesure la puissance de rayonnement :

$$\tilde{\mathcal{P}} = \int\int_{\Omega} |E_a(x)|^2 \sin\theta\ d\theta d\varphi \qquad \text{avec } x = \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \sin\theta\cos\varphi \\ \sin\theta\sin\varphi \end{pmatrix}$$

– On effectue d'abord un changement de variables pour passer des coordonnées polaires utilisées dans les définitions, au coordonnées cartésiennes utilisées dans les calculs.

On obtient :

$$\tilde{\mathcal{P}} = \int\int_{\Omega} |E_a(x)|^2 \frac{1}{w}\ dudv \qquad \text{avec} \quad w = \sqrt{1-(u^2+v^2)}$$

– On discrétise avec :

- $x_k = \begin{pmatrix} u_k \\ v_k \end{pmatrix}$ $k = \{1 \dots p\}$ l'ensemble des sommets du maillage Terre $Mesp$.
- $w_k = \sqrt{1-(u_k^2+v_k^2)}$
- $d\Lambda_k$: poids surfacique associé au point x_k, son aire, donc sa contribution au calcul de l'intégrale, va dépendre du maillage :

  Soit ah l'aire d'un hexagone de la zone de couverture de rayon R_{sp} :

  - si $x_k \in Mcs$, alors $d\Lambda_k = \frac{ah}{3}$
  - si $x_k \in Mt$, alors $d\Lambda_k = ah$

On obtient :

$$\boxed{\mathcal{P} = \sum_{k=1}^{p} |E_a(x_k)|^2 \frac{1}{w_k}\ d\Omega_k}$$

– **Notation :**

Dans le code on note $\mathcal{P} = intE2$

2.1.6 Idée clé : dériver F dans $\mathbb{R}$ par rapport à partie réelle et partie imaginaire

Pour l'algorithme d'optimisation, il faudra calculer la dériver de F.

On rappelle qu'on considère la variable a en partie réelle et partie imaginaire, afin de pouvoir dériver dans $\mathbb{R}$.

On note par la suite : $a = (x_{a_1} \dots x_{a_n} y_{a_1} \dots y_{a_n}) \in \mathbb{R}^{2n}$

avec $\forall j \; a_j = x_{a_j} + i \; y_{a_j} \quad et \quad a = (a_1 \dots a_n) \in \mathbb{C}^n$

La dérivée de F va alors s'écrire :

$$DF(a) = \begin{pmatrix} \frac{\partial F_1(a)}{\partial x_{a_1}} & \dots & \frac{\partial F_1(a)}{\partial x_{a_n}} & \frac{\partial F_1(a)}{\partial ya1} & \dots & \frac{\partial F_1(a)}{\partial yan} \\ \vdots & & \vdots & \vdots & & \vdots \\ \frac{\partial F_p(a)}{\partial x_{a_1}} & \dots & \frac{\partial F_p(a)}{\partial x_{a_n}} & \frac{\partial F_p(a)}{\partial ya1} & \dots & \frac{\partial F_p(a)}{\partial yan} \end{pmatrix} \quad \text{et} \quad DF(a) :: \mathbb{R}^{2n} \longrightarrow \mathbb{R}^p$$

2.1.7 Calcul de DF

On rencontre plusieurs cas :

- pour les sommets où les contraintes sont vérifiées ou dans Ω0

$$\forall x_j \; F_j(a) = 0 \quad \text{donc} \quad DF(a) = 0$$

- pour les sommets hors Terre où les contraintes ne sont pas vérifiées

$$\forall x_j \in \Omega_4 \backslash \Omega_3 \quad F_j(a) = g_4(a, x_j) = \frac{\iint_{\Omega_4 \backslash \Omega_3} \widetilde{\mathcal{D}}(a,x) \; dudv}{\iint_{\Omega} \widetilde{\mathcal{D}}(a,x) \; dudv} \; - 0.10$$

$$\text{donc} \quad DF_j(a) = D\left(\frac{k_1(a)}{k_2(a)}\right) = \frac{Dk_1(a)k_2(a) - k_1(a)Dk_2(a)}{k_2^2(a)}$$

avec

- $k_1(a) = \iint_{\Omega_4 \backslash \Omega_3} \widetilde{\mathcal{D}}(a,x) \; dudv$
- $k_2(a) = \iint_{\Omega} \widetilde{\mathcal{D}}(a,x) \; dudv$

Comme la fonction $\widetilde{\mathcal{D}}$ est suffisamment régulière, on peut intervertir la dérivée et l'intégrale :

$$Dk_i(a) = D \int_{\Omega_i} \widetilde{\mathcal{D}}(a) \; dudv = \int_{\Omega_i} D\widetilde{\mathcal{D}}(a) \; dudv \qquad i = 1, 2$$

- pour les tous les autres sommets où les contraintes ne sont pas vérifiées

$$F_j(a) = g^+_{I_j}(a, x_j) = \widetilde{\mathcal{D}}(a) + K \qquad \text{où K est une constante}$$

donc $DF_j(a) = D\widetilde{\mathcal{D}}(a, x_j)$

On se ramène donc au calcul de $D\widetilde{\mathcal{D}}(a, x_j)$ dans tous les cas.

2.1.8 Calcul de $D\widetilde{\mathcal{D}}(a)$

- On calcule en un sommet $X = \begin{pmatrix} u \\ v \end{pmatrix}$ fixé.
- **Notations** :

  On note

  - $\widetilde{\mathcal{D}}(a, X) = \widetilde{\mathcal{D}}(a)$ et $E(a, X) = E(a)$ par la suite.
  - $$D = \left(\frac{\partial}{\partial x_{a_1}} \cdots \frac{\partial}{\partial x_{a_n}} \vdots \frac{\partial}{\partial y_{a_1}} \cdots \frac{\partial}{\partial y_{a_n}} \right)$$
- On pose

$$\widetilde{\mathcal{D}}(a) = 4\pi \, \frac{B_1(a)}{B_2(a)}$$

avec

- $B_1(a) = |E(a)|^2 \qquad \text{et} \quad B_1 : \mathbb{R}^{2n} \longrightarrow \mathbb{R}$
- $B_2(a) = \iint_\Omega \widetilde{\mathcal{D}}(a, X) \; dudv \qquad \text{et} \quad B_2 : \mathbb{R}^{2n} \longrightarrow \mathbb{R}$

La dérivée de $\widetilde{\mathcal{D}}$ s'écrit alors :

$$D\widetilde{\mathcal{D}}(a) = \frac{4\pi(DB_1(a)B_2(a) - B_1(a)DB_2(a))}{B_2^2(a)}$$

Les calculs de $DB_1(a)$ et $DB_1(a)$ sont présentés dans la suite.

2.2 Calcul de la dérivée de la directivité $D\widetilde{\mathcal{D}}(a, x)$

La dérivée de $\widetilde{\mathcal{D}}$ s'écrit :

$$D\widetilde{\mathcal{D}}(a) = \frac{4\pi(DB_1(a)B_2(a) - B_1(a)DB_2(a))}{B_2^2(a)}$$

2.2.1 Calcul de $DB_1(a)$:

On simplifie d'abord l'expression.

$$|E(a)|^2 = \left| \sum_{j=1}^{n} a_j \; e^{i\frac{2\pi}{\lambda}\langle \overrightarrow{OM_j}, \overrightarrow{OX} \rangle} \; E_j \right|^2$$

$$|E(a)|^2 = \left| \sum_{j=1}^{n} (x_{a_j} + iy_{a_j}) \; (\cos(expo_j) + i \; \sin(expo_j)) \, E_j \right|^2$$

en posant

– $expo_j := \left(\frac{2\pi}{\lambda}\langle\overrightarrow{OM_j},\overrightarrow{OX}\rangle\right)$

$$|E(a)|^2 = \left|\left(\sum_{j=1}^{n}\ \left[x_{a_j}\cos(expo_j) - y_{a_j}sin(expo_j)\right] +\ i\ \left[x_{a_j}\sin(expo_j) + y_{a_j}\cos(expo_j)\right]\right) E_j\ \right|^2$$

$$|E(a)|^2 = \left|\sum_{j=1}^{n}\left(C_1(a_j)E_j + i\ C_2(a_j)E_j\right)\right|^2$$

en posant

– $C_2(a_j) := \left(x_{a_j}\sin(expo_j) + y_{a_j}\cos(expo_j)\right)$

– $C_1(a_j) := \left(x_{a_j}\cos(expo_j) - y_{a_j}\sin(expo_j)\right)$

$$|E(a)|^2 = \left|\sum_{j=1}^{n} C_1(a_j)E_j + i\ \sum_{j=1}^{n} C_2(a_j)E_j\right|^2$$

$$|E(a)|^2 = \left|\underbrace{\left(\sum_{j=1}^{n} C_1(a_j)E_j\right)}_{\in\mathbb{R}} + i\ \underbrace{\left(\sum_{j=1}^{n} C_2(a_j)E_j\right)}_{\in\mathbb{R}}\right|^2$$

$$|E(a)|^2 = \left(\sum_{j=1}^{n} C_1(a_j)E_j\right)^2 +\ \left(\sum_{j=1}^{n} C_2(a_j)E_j\right)^2$$

On dérive l'expression précédente.

$$DB_1(a) = D\left(|E(a)|^2\right) = D\left[\left(\sum_{j=1}^{n} C_1(a_j)E_j\right)^2\right] +\ D\left[\left(\sum_{j=1}^{n} C_2(a_j)E_j\right)^2\right]$$

$$DB_1(a) = 2\ \left(\sum_{j=1}^{n} C_1(a_j)E_j\right)\ D\left[\sum_{j=1}^{n} C_1(a_j)E_j\right] + 2\ \left(\sum_{j=1}^{n} C_2(a_j)E_j\right)\ D\left[\sum_{j=1}^{n} C_1(a_j)E_j\right]$$

$$DB_1(a) = 2\ \left(\sum_{j=1}^{n} C_1(a_j)E_j\right)\ \left(\sum_{j=1}^{n} DC_1(a_j)\ E_j\right) + 2\ \left(\sum_{j=1}^{n} C_2(a_j)E_j\right)\ \left(\sum_{j=1}^{n} DC_2(a_j)\ E_j\right)$$

avec

- $C_1(a_j) = \left(x_{a_j}\cos(expo_j) - y_{a_j}\sin(expo_j)\right)$
- $$DC_1(a_j) = \left(\frac{\partial}{\partial x_{a_1}} \dots \frac{\partial}{\partial x_{a_n}} \vdots \frac{\partial}{\partial y_{a_1}} \dots \frac{\partial}{\partial y_{a_n}}\right) C_1(a_j)$$

  donc $DC_1(a_j) = (\dots 0 \dots \underbrace{\cos(expo_j)}_{\text{indice } i} \dots 0 \dots \vdots \dots 0 \dots \underbrace{-\sin(expo_j)}_{\text{indice } n+i} \dots 0 \dots)$
- de même $DC_2(a_j) = (\dots 0 \dots \underbrace{\sin(expo_j)}_{\text{indice } i} \dots 0 \dots \vdots \dots 0 \dots \underbrace{\cos(expo_j)}_{\text{indice } n+i} \dots 0 \dots)$

Finalement on obtient :

$DB_1(a) =$

$$2\left(\sum_{j=1}^{n} C_1(a_j)E_j\right)\left(cos(expo_1)E_1 \dots cos(expo_n)E_n \vdots -sin(expo_1)E_1 \dots -sin(expo_n)E_n\right)$$

$$+2\left(\sum_{j=1}^{n} DC_2(a_j)\ E_j\right)\left(sin(expo_1)E_1 \dots sin(expo_n)E_n \vdots -cos(expo_1)E_1 \dots -cos(expo_n)E_n\right)$$

2.2.2 Calcul de $DB_2(a)$:

Comme la fonction $\widetilde{\mathcal{D}}$ est sufisamment régulière, on peut intervertir l'intégrale et la dérivée :

$$DB_2(a) = D\iint_\Omega \widetilde{\mathcal{D}}(a,X)\ dudv = \iint_\Omega D\widetilde{\mathcal{D}}(a,X)\ dudv$$

On retrouve le calcul de $D\widetilde{\mathcal{D}}(a)$, il faudra ensuite discrétiser l'intégrale de la même manière que pour pour le calcul de la puissance (voir Calcul de la puissance au Chapitre 2).

Auteur: Thierry TOUYA
Titre: Méthodes d'optimisation pour l'espace et l'environnement
Directeurs: Didier AUROUX et Mohamed MASMOUDI
Thèse soutenue le 26/09/2008 à l'Institut de Mathématiques de Toulouse

---

Titre: Méthodes d'optimisation pour l'espace et l'environnement

Résumé:
Ce travail se compose de deux parties relevant d'applications industrielles différentes.
La première traite d'une antenne spatiale réseau active.
Il faut d'abord calculer les lois d'alimentation pour satisfaire les contraintes de rayonnement. Nous transformons un problème avec de nombreux de minima locaux en un problème d'optimisation convexe, dont l'optimum est le minimum global du problème initial, en utilisant le principe de conservation de l'énergie.
Nous résolvons ensuite un problème d'optimisation topologique: il faut réduire le nombre d'éléments rayonnants (ER). Nous appliquons une décomposition en valeurs singulières à l'ensemble des modules optimaux relaxés, puis un algorithme de type gradient topologique décide les regroupements entre ER élémentaires.

La deuxième partie porte sur une simulation type boîte noire d'un accident chimique.
Nous effectuons une étude de fiabilité et de sensibilité suivant un grand nombre de paramètres (probabilités de défaillance, point de conception, et paramètres influents). Sans disposer du gradient, nous utilisons un modèle réduit.
Dans un premier cas test nous avons comparé les réseaux neuronaux et la méthode d'interpolation sur grille éparse *Sparse Grid* (SG). Les SG sont une technique émergente: grâce à leur caractère hiérarchique et un algorithme adaptatif, elles deviennent particulièrement efficaces pour les problèmes réels (peu de variables influentes).
Elles sont appliquées à un cas test en plus grande dimension avec des améliorations spécifiques (approximations successives et seuillage des données).

Dans les deux cas, les algorithmes ont donné lieu à des logiciels opérationnels.

Mots clés: Antenne réseau active - Synthèse de réseau - Optimisation convexe - Optimisation topologique - Fiabilité - Sensibilité - Probabilités de défaillance - Point de conception - Réseaux de neurones - Sparse Grids

---

Title: Optimization methods for space and environment

Abstract:
This work is composed of two parts, issued from different industrial applications.
The first one is about an active array antenna.
First, we have to calculate the optimal excitations in order to comply with the radiated power requirements. We transform a problem with numerous local minima into a convex optimization one, whose optimum is the global minimum of the initial problem, by using the energy conservation principle.
Then we solve a topological optimization problem: we have to reduce the number of Radiating Elements (RE). We apply a singular value decomposition to all relaxed optimal modulus, and a topological gradient algorithm decides gatherings of elementary RE.

The second part is about a chemical accident black box modelling.
We perform a reliability and sensitivity analysis of a large number of parameters (failure probabilities, design point, and influent parameters). Without the gradient, we use a reduced model.
In a first test case, we compare the neural networks and the Sparse Grids (SG) interpolation method. The SG are an emergent tool: thanks to their hierarchical structure and an adaptive algorithm, they become particularly efficient for real problems (few influent variables).
We apply them to a test case in larger dimension with specific improvements (successive approximations and data thresholds). In both cases, algorithms have lead to operative software.

Key-words: Active array antenna - Convex optimization - Topological optimization
Reliability - Sensitivity - Failure probabilities - Design point - Neural networks - Sparse Grids